Stimate cumpărător,

The COMPLETE SPECTUM ROM DISASSEMBLY (ROM-ul complet dezasamblat) este o carte unanim acceptată ca fiind manualul de referință pentru toți utilizatorii serioși ai calculatoarelor Spectrum și compatibile. Cartea lui Dr. Ian Logan & Dr.Frank O'Hara, scoasă la editura Melbourne House Publishers este tot ceea ce poate visa un utilizator de Spectrum & compatibile pentru a obține cît mai mult din sistemul său. La această lucrare firma noastră nu urmăreste decît popularizarea unei lucrări absolut remarcabile (am listat cu 70 de caractere pe rînd peste 300 de pagini, compact, la cel mai scăzut preț posibil) pentru a ajuta la înțelegerea cît mai detaliată a acestei minunate unelte care este calculatorul.

"ALPHA Ltd."

May 12, 1991
6:35 PM

CUPRINS

	Pag.
Prefata	
Introducere	4
ASAMBLORUL	6
- Rutinele de restart si tabele	6
- Rutinele tastaturii	11
- Rutinele de difuzor	19
- Rutinele de tratare a casetei	24
- Rutinele de tratare a ecranului si imprimantei	49
- Rutinele executivului	84
- Programul BASIC si interpretarea comenzilor	116
- Evaluarea expresiei	172
- Rutinele aritmetice	222
- Dispozitivul de calcul in virgula mobila	259
Anexa	
- Programele BASIC pentru seriile principale (SIN X, EXP X, LN X & ATN X)	302
- Algoritmul 'DRAW' (desenare)	308
- Algoritmul 'CIRCLE' (cerc)	308
- Nota asupra intregilor mici si asupra lui -65536	310
Indexul rutinelor	312

PREFATA

Spectrum ZX Sinclair este un succesor merituos al lui ZX 81 care la rîndul înlocuieste ZX 80.

Spectrum are un program monitor de 16 kocteti. Acest program a fost dezvoltat direct din programul monitor de 4 koctati al lui ZX 80, desi aici sînt atît de multe trăsături noi încît diferentele prevalează asupra asemănărilor.

Am avut o deosebită plăcere elaborînd această carte. Am învătat multe lucrînd asupra tehnicilor de programare în cod masină Z80 si acum simtim că am reusit să lămurim 'secretele Spectrum'.

Am vrea să multumim:
- Familiilor noastre
- Lui Alfred Milgrom, editorul nostru care ne-a fost de mare ajutor
- lui Philip Mitchell, ale cărui notări asupra formării casetei ne-au adus multe informatii
- lui Clive Sinclair si echipei lui si lui Sinclair Research Ltd., care au realizat o masină atît de folositoare si incitantă.

Ianuarie 1983.

INTRODUCERE

Programul monitor de 16 k al lui Spectrum este un program scris în cod mașină Z80 complex. Structura sa generală este foarte clară în împărțirea sa în trei mari părți:

a. Rutine de Intrare/Ieșire
b. Interpretorul BASIC
c. Tratarea expresiei

Totuși aceste blocuri sînt prea întinse pentru a fi descurcate ușor și de aceea în această carte programul monitor este discutat în zece părți.
Fiecare din aceste părți va fi acum 'achitată'.

Rutinele de restart și tabele

La începutul programului monitor sînt diferitele rutine de 'restart' care sînt apelate cu ajutorul instrucțiunilor pe un singur octet 'RST'. Toate 'repornirile' sînt folosite. De exemplu, 'restart 0008' este folosit pentru raportările sintaxei sau a erorilor de execuție.
Tabelele din această parte a programului monitor conțin formele extinse ale simbolurilor și 'codurile tastelor'.

Rutinele tastaturii

Tastatura este baleiată la fiecare a 1/50-a parte dintr-o secundă (model european) și rutinele tastaturii redau codul caracterului cerut. Toate tastele claviaturii autorepetă dacă sînt apăsate și acest lucru este luat în considerare de către rutinele tastaturii.

Rutinele de difuzor

Spectrum are încorporat un singur difuzor și o notă este produsă prin folosirea repetată a instrucțiunii corespunzătoare 'OUT'. In rutina controloare se are mare grijă să se asigure că nota este păstrată la o înălțime dată pe toată 'durata' sa.

Rutinele de tratare a casetei

A fost o foarte nefericită trăsătură a lui ZX 81 că așa de puțin din programul monitor pentru această mașină a fost consacrat tratării casetei.
In orice caz în Spectrum există un bloc extins al codului și acum standardul înalt de tratare a casetei este una din cele mai reușite trăsături ale mașinii.
Programele BASIC sau blocurile de informații lucrează amîndouă în același fel, avînd un bloc 'header' (17 octeți) care este salvat mai întîi. Acest 'header' descrie 'blocul de informații' care este salvat după el.
Un dezavantaj al acestui sistem este faptul că nu este posibilă realizarea de programe cu o oarecare 'siguranță'.

Rutinele de tratare a ecranului și imprimantei

Toate rutinele de intrare/ieșire rămase ale lui Spectrum sînt 'orientate' direct în 'canalul 4 zona șirului de informații'.
In standardul Spectrum 'intrarea' este posibilă doar de la tastatură, dar 'ieșirea' poate fi direcționată la imprimantă, în partea superioară a ecranului TV sau în partea inferioară a ecranului TV.
Principala rutină 'de ieșire' din această parte a programului monitor este EDITOR, care permite utilizatorului introducerea de caractere în partea inferioară a ecranului TV.
Rutina PRINT-OUT este o rutină destul de întreagă care este folosită pentru 'toate posibilitățile'. De exemplu: adăugarea unui singur octet în 'spațiul ecranului' implică luarea în considerare a stării prezente a lui OVER și INVERSE în fiecare ocazie.

Rutinele executivului

In această parte a programului monitor trebuie găsită procedura INITIALISATION și 'bucla principală de execuție' a interpretorului BASIC.
In Spectrum linia BASIC returnată de către EDITOR este verificată

din punct de vedere al corectitudinii sintaxei si apoi salvatà în spatiul programului, dacà a fost o linie care a început cu un numàr de linie.

Linia BASIC si interpretarea comenzii

Această parte a programului monitor consideră linia BASIC ca un set de exprimări, fiecare exprimare la rîndul ei începînd cu o comandă particulară. Pentru fiecare comandă există o 'rutina de comandă', iar interpretarea este efectuată de executia codului masină în 'rutina de comandă' potrivită.

Evaluarea expresiei

Spectrum are cea mai cuprinzătoare evaluare de expresie, disponibilă pentru o gamă largă de tipuri de variabile, functii, operatii.
Tratarea sirurilor este, în mod particular, bine rezolvată. Toate sirurile simple sînt rezolvate 'dinamic', iar rezultatele cunoscute sînt 'recuperate' odată ce sînt redundante. Aceasta înseamnă că nu este de făcut nici o 'curătenie'.

Rutinele aritmetice

Spectrum are două forme pentru numere. Valorile din domeniul -65535 ÷ +65535 sînt în formă 'integrală' sau 'scurtă' pe cînd toate celelalte numere sînt în formă de 5 biti cu virgulă mobilă.
Versiunea de fată a monitorului are în această privintă, din păcate, două inconveniente:
i. Există un inconvenient la împărtire prin faptul că se pierde al 34-lea bit.
ii. Valoarea lui -65536 este cîteodată pusă în forma 'scurtă' iar în alte dăti în 'virgulă mobilă', ceea ce conduce la necazuri.

Calcularea virgulei mobile

CALCULATORUL din Spectrum tratează numerele si sirurile, iar operatiile lui sînt specificate prin 'litere'. De aceea se poate considera că există un limbaj de operare intern în CALCULATOR.
Această parte a programului monitor contine rutine pentru toate functiile matematice. Aproximatiile la SIN X, EXP X, LN X si respectiv ATN X sînt obtinute dezvoltînd polinoamele Chebyshev, mai multe detalii fiind date în appendix.

In totalitate, programul monitor de 16K, oferă o gamă extrem de largă de diferite comenzi si functii BASIC. Cu toate acestea, programatorii întotdeauna duc lipsă de 'posibilitate' si din acest motiv programul este scris mai degrabă pentru 'scurtime decît pentru 'viteză'.

Rutinele dezasamblate

RUTINELE RESTART si TABELELE

THE START (START)
Intreruperea mascabila este dezactivata si registrul pereche DE este fixat sa pastreze 'vîrful RAM-ului posibil'.

```
0000 START      DI                          Dezactiveaza 'intrerupere
                XOR   A,+00                 claviatura' pentru 'start'
                                            (dar +FF pentru 'NEW')
                LD    DE,+FFFF              Vîrful RAM-ului posibil.
                JP    11CB,START/NEW        Salt în față.
```

THE 'ERROR' RESTART
Pointerul de eroare este făcut să indice poziția erorii.

```
0008 ERROR-1    LD    HL,(CH-ADD)           Adresa ajunsă de interpretor
                LD    (X-PTR),HL            este copiată la pointerul de
                JR    0053,ERROR-2          eroare înainte de a continua.
```

THE 'PRINT A CHARACTER' RESTART ('TIPARESTE UN CARACTER')
Registrul A păstrează codul caracterului care este tipărit.

```
0010 PRINT-A-1  JP    5F2,PRINT-A-2         Salt în față imediat.
                DEFB  FF,+FF,+FF,+FF,+FF    Locatii nefolosite.
```

THE 'COLLECT CHARACTER' RESTART ('CULEGE CARACTER')
Sînt scoase continuturile locației adresată curent de CH-ADD. Se face o întoarcere dacă valoarea reprezintă un caracter ce se poate tipări, în caz contrar CH-ADD este incrementat si testele sînt repetate.

```
0018 GET-CHAR   LD    HL,(CH-ADD)           Scoate valoarea care este
                LD    A,(HL)                adresată de CH-ADD.
0010 TEST-CHAR  CALL  007D,SKIP-OVER        Constată dacă caracterul se
                RET                         poate tipări. Dacă este
                                            așa, întoarcere.
```

THE 'COLLECT NEXT CHARACTER' RESTART ('CULEGE URMATORUL CARACTER')
Cum o linie BASIC este interpretabilă, această rutină este chemată frecvent de-a lungul liniei.

```
0020 NEXT-CHAR  CALL  0074,CH-ADD+1         CH-ADD trebuie să fie
                                            incrementat.
                JR    001C,TEST-CHAR        Salt înapoi pentru a testa
                                            noua valoare.
                DEFB  +FF,+FF,+FF           Locatii nefolosite.
```

THE 'CALCULATOR' RESTART (RESTART 'CALCULATOR')
Virgula mobilă este introdusă la 335B.

```
0028 FP-CALC    JP    335B,CALCULATE        Salt neconditionat în față.
                DEFB  +FF,+FF,+FF,+FF,+FF   Locatii nefolosite.
```

THE 'MAKE BC SPACES' RESTART ('CREEAZA SPATII BC')
Această rutină creează locații libere în spațiul de lucru. Numărul locațiilor este determinat de conținuturile curente ale registrului pereche BC.

```
0030 BC-SPACES  PUSH  BC                    Salvează 'numărul'.
                LD    HL,(WORKSP)           Scoate adresa prezentă a
                PUSH  HL                    începutului spațiului de
                JP    169E,RESERVE          lucru si salveaz-o înainte de
                                            a merge mai departe.
```

THE 'MASKABLE INTERRUPT' ROUTINE ('INTRERUPERE MASCABILA')
Ceasul de timp real este incrementat, iar tastatura cercetată pentru a sesiza cînd apare o întrerupere mascabilă.

```
0039 MASK-INT   PUSH  AF                    Salvează valorile curente
                PUSH  HL                    memorate în aceste registre
                LD    HL,(FRAMES)           Cei mai puțin semnificativi 2
                LD    (FRAMES),HL           biți ai fiecărei 20 ms (U.K.)
                LD    A,H                   Cel mai semnificativ bit al
                OR    L                     numărătorului de cadre este
                JR    NZ,0048,KEY-INT       incrementat numai cînd
                INC   (FRAMES-3)            valoarea celor doi biți mai
```

```
                                                    putin semnificativi este
                                                    zero.
0048 KEY-INT    PUSH  BC            Salveaza valorile curente
                PUSH  DE            pastrate in aceste registre
                CALL  02BF,KEYBOARD Acum cerceteaza tastatura.
                POP   DE            Se refac valorile.
                POP   BC
                POP   HL
                POP   AF
                EI                  Intreruperea mascabila este
                RET                 permisa inainte de
                                    intoarcere.
```

THE 'ERROR-2' ROUTINE
Adresa de intoarcere la interpretor indica la 'DEFB' ca, semnifica ce eroare a aparut. Acest 'DEFB' este scos si transferat la ERR-NR. Stiva masinii este stearsa inainte de a sari mai departe pentru a sterge stiva calculatorului.

```
0053 ERROR-2    POP   HL            Adresa din stiva semnifica
                LD    L,(HL)        codul erorii.
0055 ERROR-3    LD    (ERR-NR),L    El este transferat la ERR-NR.
                LD    SP,(ERR-SP)   Masina este stearsa
                JP    16C5,SET-STK  inainte de a iesi prin
                                    SET-STK.
                DEFB  +FF,+FF,+FF,+FF Locatii nefolosite.
                DEFB  +FF,+FF,+FF
```

THE 'NON-MASKABLE INTERRUPT' ROUTINE (RUTINA 'INTRERUPERE NEMASCABILA')
Aceasta rutina nu este folosita in Spectrum-ul standard, dar urmarind activarea liniei NMI codul permite aparitia unui reset sistem. Variabila sistem la 5CB0, numita aici NMIADD, trebuie sa aiba valoarea zero pentru a apare un reset.

```
0066 RESET      PUSH  AF            Salveaza valorile curente
                PUSH  HL            pastrate in aceste registre.
                LD    HL,(NMIADD)   Cei doi biti ai NMIADD
                LD    A,H           trebuie sa fie ambii zero
                OR    L             pentru ca reset-ul sa apara.
                JR    NZ,0070,NO-RESET Nota: Aceasta ar fi
                                    trebuit sa fie 'JR Z'!
                JP    (HL)          Salt la START.
0070 NO-RESET   POP   HL            Se refac valorile curente
                POP   AF            la aceste registre si
                RET   N             intoarcere.
```

THE 'CH-ADD+1' SUBROUTINE
Adresa pastrata in CH-ADD este scoasa, incrementata si reintrodusa. Este scos continutul locatiei adresata acum de CH-ADD. Punctele de intrare TEMP-PTR1 si TEMP-PTR2 sint folosite pentru a fixa CH-ADD pentru o perioada temporara.

```
0074 CH-ADD+1   LD    HL,(CH-ADD)   Scoate adresa.
0077 TEMP-PTR1  INC   HL            Se incrementeaza pointerul.
0078 TEMP-PTR2  LD    (CH-ADD),HL   Fixeaza CH-ADD.
                LD    A,(HL)        Scoate valoarea adresata si
                RET                 apoi intoarcere.
```

THE ' SKIP-OVER' SUBROUTINE (SUBRUTINA 'SARE PESTE')
Valoarea adusa subrutinei in registrul A este testata pentru a vedea daca se poate tipari. Diferite coduri speciale conduc ca HL sa fie incrementat o data, sau de doua ori, si CH-ADD corectat corespunzator.

```
007D SKIP-OVER  CP    +21           Intoarcere cu indicatorul de
                RET   NC            stare carry resetat.
                CP    +0D           Intoarcere daca s-a ajuns
                RET   Z             la capatul liniei.
                CP    +10           Intoarcere cu codurile +00 la
                RET   C             +0F dar cu indicatorul de
                                    stare fixat.
                CP    +18           Intoarcere cu codurile +18 la
                CCF                 +20 din nou cu indicatorul
                RET   C             carry fixat.
                INC   HL            Incrementeaza inca o data
                CP    +16           Salt in fata cu codurile +10
                JR    C,0090,SKIPS  la +15 (de la INK la OVER).
                INC   HL            Incrementeaza inca o data
                                    (AT&TAB).
0090 SKIPS      SCF                 Intoarcere cu indicatorul
                LD    (CH-ADD),HL   carry fixat si cu CH-ADD
                RET                 pastrind adresa potrivita.
```

THE TOKEN TABLE (TABELUL DE SIMBOLURI)
Toate simbolurile folosite de Spectrum sînt dezvoltări avînd ca referință acest tabel. Ultimul cod al fiecărui simbol 'inversat' are bitul său 7 fixat.

0095	BF	52	4E	C4	49	4E	4B	45	'?'	R	N	'D'	I	N	K	E
009D	59	A4	50	C9	46	CE	50	4F	Y	'$'	P	'I'	F	'N'	P	O
00A5	49	4E	D4	53	43	52	45	45	I	N	'T'	S	C	R	E	E
00AD	4E	A4	41	54	54	D2	41	D4	N	'$'	A	T	T	'R'	A	'T'
00B5	54	41	C2	56	41	4C	A4	43	T	A	'B'	V	A	L	'$'	C
00BD	4F	44	C5	56	41	CC	4C	45	O	D	'E'	V	A	'L'	L	E
00C5	CE	53	49	CE	45	4F	D3	54	'N'	S	I	'N'	E	O	'S'	T
00CD	A1	CE	A1	33	CE	A1	43	D3	A	'N'	A	S	'N'	A	C	'S'
00D5	A1	54	CE	4C	CE	45	58	D0	A	T	'N'	L	'N'	E	X	'P'
00DD	49	4E	D4	53	51	D2	53	47	I	N	'T'	S	Q	'R'	S	G
00E5	CE	41	42	D3	50	45	45	CB	'N'	A	B	'S'	P	E	E	'K'
00ED	49	CE	55	53	D2	53	54	52	I	'N'	U	S	'R'	S	T	R
00F5	A4	43	48	52	A4	4E	4F	D4	'$'	C	H	R	'$'	N	O	'T'
00FD	42	49	CE	4F	D2	41	4E	C4	B	I	'N'	O	'R'	A	N	'D'
0105	3C	BD	3E	BD	3C	BE	4C	49	<	=	>	<>	<=	>=	L	I
010D	4E	C5	54	48	45	CE	54	CF	N	'E'	T	H	E	'N'	T	'O'
0115	53	54	45	D0	44	45	46	20	S	T	E	'P'	D	E	F	
011D	46	4E	43	41	D4	46	4F	52	F	'N'	C	A	'T'	F	O	R
0125	4D	4F	D6	4D	4F	56	C5	45	M	A	'T'	M	O	V	'E'	E
012D	52	41	53	C5	4F	50	45	4E	R	A	S	'E'	O	P	E	N
0135	20	A3	43	AC	4F	53	45	20		'#'	C	L	O	S	E	
013D	A3	4D	45	52	47	C5	54	45	'#'	M	E	R	G	'E'	V	E
0145	52	49	46	D9	42	45	45	D0	R	I	F	'Y'	B	E	E	'P'
014D	43	49	52	43	4C	C5	49	4E	C	I	R	C	L	'E'	I	N
0155	CB	50	41	50	45	D2	46	4C	'K'	P	A	P	E	'R'	F	L
015D	41	53	C8	42	52	49	47	48	A	S	'H'	B	R	I	G	H
0165	D4	49	4E	56	45	52	53	C5	'T'	I	N	V	E	R	S	'E'
016D	4F	56	45	D2	4F	55	D4	4C	O	V	E	'R'	O	U	'T'	L

0175	50	52	49	4E	D4	4C	4C	49	P	R	I	N	'T'	L	L	I
017D	53	B4	53	54	4F	B0	52	45	S	'T'	S	T	O	'P'	R	E
0185	41	C4	44	41	54	C1	B2	43	A	'D'	D	A	T	A	R	E
018D	33	54	4F	52	C5	4E	45	B7	3	T	O	R	'E'	N	E	'W'
0195	42	4F	B2	44	45	B2	43	4F	B	O	R	D	E	'R'	C	O
019D	4E	54	49	4E	55	C5	44	49	N	T	I	N	U	'E'	D	I
01A5	CD	52	45	CF	46	4F	D2	47	'M'	R	E	'M'	F	O	'R'	G
01AD	4F	20	54	CF	47	4F	20	53	O		T	'O'	O	O		S
01B5	55	C2	49	4E	50	55	B4	4C	U	'B'	I	N	P	U	'T'	L
01BD	4F	41	C4	4C	49	53	D4	4C	O	A	'D'	L	I	S	'T'	L
01C5	45	B4	50	41	55	53	C5	4E	E	'T'	P	A	U	S	'E'	N
01CD	45	58	B4	50	4F	4B	C5	50	E	X	'T'	P	O	K	'E'	P
01D5	52	49	4E	B4	50	4C	4F	B4	R	I	N	'T'	P	L	O	'T'
01DD	52	55	CE	53	41	56	C5	52	R	U	'N'	S	A	V	'E'	R
01E5	41	4E	44	4F	4B	49	5A	C5	A	N	D	O	K	I	Z	'E'
01ED	49	C6	43	4C	D3	44	52	41	I	'F'	C	L	'S'	D	R	A
01F5	B7	43	4F	45	41	D2	52	45	'W'	C	L	E	A	'R'	R	E
01FD	54	55	52	CE	43	4F	50	B9	T	U	R	'N'	C	O	P	'Y'

THE KEY TABLES (TABELE REFERITOARE LA TASTE)

Există șase tabele diferite. Codul final al caracterului obținut depinde de tasta care a fost apăsată și de 'modul' folosit.

( )aTabelul principal de taste - Modul L și CAPS SHIFT

0205	42	48	58	36	35	54	47	56		B		H		Y	6	5	T	G	V
020D	4E	4A	55	37	34	52	46	43		N		J		U	7	4	R	F	C
0215	4D	4B	49	38	33	45	44	58		M		K		I	8	3	E	D	K
021D	0E	4C	4F	39	32	57	53	5A		SYMBOL SHIFT		L		O	9	2	W	S	Z
0225	20	0D	50	30	31	51	41			SPACE		ENTER		P	0	1	Q	A	

( )bModul extins. Taste literale si nesiftate.

022C	E3	C4	E0	E4	READ	BIN	LPRINT	DATA
0230	B4	BC	BD	BB	TAN	SGN	ABS	SQR
0234	AF	B0	B1	C0	CODE	VAL	LEN	USR
0238	A7	A6	BE	AD	PI	INKEY$	PEEK	TAB
023C	B2	BA	E5	A5	SIN	INT	RESTORE	RND
0240	C2	E1	B3	B9	CHR$	LLIST	COS	EXP
0244	D1	B8			STR$	LN		

( )cModul extins. Taste literale si siftate.

0246	7E	DC	DA	5C	/\/	BRIGHT	PAPER	\
024A	B7	7B	7D	DB	ATN	{	}	CIRCLE
024E	BF	AE	AA	AB	IN	VAL$	SCREEN$	ATTR
0252	BD	DE	DF	7F	INVERSE	OVER	OUT	©
0256	B5	B6	7C	B5	ASN	VERIFY	|	MERGE
025A	5D	DB	B6	D9	]	FLASH	ACS	INK
025E	5B	B7			[	BEEP		

( )dCoduri de control. Taste digitale si CAPS SHIFT.

0260	0C	07	06	04	DELETE	EDIT	CAPS LOCK	TRUE VIDEO
0264	05	08	0A	0B	INV VIDEO	Cursor left	Cursor down	Cursor up
0268	09	0f			Cursor right	GRAPHICS		

( )eCoduri simbol. Taste literale si simbolul shift.

026A	E2	2A	3F	CD	STOP	*	?	STEP
026E	C8	CC	CB	5E	>=	TO	THEN	
0272	AC	2D	2B	3D	AT	-	+	=
0276	2E	2C	3B	22	.	,	;	"
027A	C7	3C	C3	3E	<=	<	NOT	>
027E	C5	2F	C9	60	OR	/	<>	£
0282	C6	3A			AND	:		

( )fModul extins. Taste digitale si simbolul shift.

0284	B0	CE	A8	CA	FORMAT	DEF FN	FN	LINE
0288	B3	D4	D1	D2	OPEN	CLOSE	MOVE	ERASE
028C	A9	CF			POINT	CAT		

RUTINELE TASTATURII

THE 'KEYBOARD SCANNING' SUBROUTINE (SUBRUTINA 'CERCETARE TASTATURA')
Această subrutină foarte importantă este apelată atît de principala subrutină a tastaturii cît și de rutina INKEY$ (în SCANNING).
În toate cazurile registrul E este reschimbat cu o valoare în domeniul de la +00 la +27, valoarea fiind diferită pentru fiecare dintre cele 40 de chei ale tastaturii, sau valoarea +FF, pentru nici o cheie.

Registrul D este neschimbat cu o valoare ce indică care cheie shift singulară este apăsată. Dacă ambele chei shift sînt apăsate atunci registrele D și E sînt neschimbate cu valorile pentru CAPS SHIFT și respectiv pentru cheia SYMBOL SHIFT.
Dacă nu s-a apăsat nici o tastă, atunci registrul pereche DE va păstra neschimbat +FFFF.
Indicatorul zero este resetat dacă mai mult de două taste sînt apăsate sau dacă nici una din tastele unei perechi de taste nu este o tastă shift.

```
028E KEY-SCAN   LD   L,+2F        Valoarea inițială a cheii pentru fiecare linie va fi
                                  +2F, +2E..., +28 (8 linii).
                LD   DE,+FFFF     Inițializează DE pentru 'nici o tastă'.
                LD   BC,+FEFE     C = port de adrese,
                                  B = numărător
```

Acum intră într-o buclă. Se fac opt treceri, la fiecare pas tasta avînd o valoare inițială diferită și se va cerceta o linie diferită de 5 taste. (Prima linie este CAPS SHIFT, Z, X, C, V).

```
0296 KEY-LINE   IN   A,(C)        Citește de la portul specificat.
                CPL               O tastă apăsată în linie va
                AND  +1F          fixa bitul său respectiv
                                  (de la bit 0 - tastă
                                  exterioară, la bit 4 - tastă
                                  internă).
                JR   Z,02AB,KEY-DONE  Salt în față dacă nici una
                                  din cele 5 taste nu a fost
                                  apăsată.
                LD   H,A          Biții tastei merg în
                LD   A,L          registrul H în timp ce este
                                  adusă valoarea inițială a
                                  tastei.
029F KEY-3KEYS  INC  D            Dacă sînt apăsate trei taste
                RET  NZ           pe tastatură, atunci
                                  registrul D nu va mai menține
                                  +FF;dacă aceasta se întîmplă,
                                  întoarcere.
02A1 KEY-BIT    SUB  +08          Frecvent se scade '8' din
                SRL  H            valoarea prezentă a tastei
                JR   NC,02A1,KEY-BITS  până se găsește un key-bit.
                LD   D,E          Copiază oricare primă valoare
                                  de tastă în registrul D.
                LD   E,A          Pasează noua valoare a tastei
                                  în registrul E.
                JR   NZ,029F,KEY.3KEYS  Dacă există în această linie
                                  o a 2-a sau o a 3-a posibilă
                                  tastă apăsată, atunci salt
                                  înapoi.
02AB KEY-DONE   DEC  L            Linia a fost cercetată astfel
                                  că valoarea inițială a cheii
                                  este redusă pentru pasul
                                  următor.
                RLC  B            Numărătorul este shiftat și
                JR   C,0296,KEY-LINE  săritura are loc dacă mai
                                  sînt linii de cercetat.
```

Acum se mai fac patru teste.

```
                LD   A,D          Acceptă orice valoare de
                INC  A            tastă care totuși menține
                RET  Z            în registrul D, +FF. Adică o
                                  singură tastă apăsată sau
                                  'nici o tastă'.
                CP   +28          Acceptă valoarea tastei
                RET  Z            pentru o pereche de taste
```

```
                                    dacă tasta 'B' este CAPS
                                    SHIFT.
         CP    +19                  Acceptă valoarea tastei
         RET   Z                    pentru o pereche de
                                    taste dacă tasta 'B' este
                                    SYMBOL SHIFT.
         LD    A,E                  Este totuși posibil pentru
         LD    E,D                  tasta 'E' a unei perechi să
         LD    D,A                  fie SYMBOL SHIFT - astfel
         CP    +18                  aceasta trebuie luată în
                                    considerare.
         RET                        Întoarcere cu indicatorul
                                    zero fixat dacă a fost SYMBOL
                                    SHIFT și 'o altă tastă';
                                    altfel este resetat.
```

THE 'KEYBOARD' SUBROUTINE (TASTATURA)

Această subrutină este chemată în fiecare ocazie în care apare o întrerupere mascabilă. În operații normale aceasta se întîmplă o dată la fiecare 20 ms. Scopul acestei subrutine este de a cerceta tastatura și de a codifica valoarea tastei. Acest cod produs - dacă starea de 'repetare' o permite - va fi trecut la variabila sistem LAST-K. Cînd un cod este pus în această variabilă sistem, bitul 5 al FLAG3 este fixat să arate că o 'nouă' tastă a fost apăsată.

```
02BF KEYBOARD   CALL  028E,KEY-SCAN    Se aduce o valoare de tastă
                RET   NZ               în registrul pereche DE cu
                                       întoarcere imediată dacă
                                       indicatorul zero este
                                       resetat.
```

De aici încolo se va folosi un sistem dublu de 'variabile sistem KSTATE' (KSTATE0 - KSTATE3 SI KSTATE4 - KSTATE7).
Cele două seturi țin seama de detectia unei noi taste care este apăsată (folosind un singur set) încă în timpul 'perioadei de repetare' a tastei ce a fost anterior apăsată (detalii în al 2-lea set).
Un set va deveni liber să tasteze o nouă tastă numai dacă este apăsată aproximativ 1/10 părți dintr-o secundă, adică 5 cereri la tastatură.

```
                LD    HL,KSTATE0       Incepe cu KSTATE0.
02C6 K-ST-LOOP  BIT   7,(HL)           Salt mai departe dacă un set
                JR    NZ,02D1,K-CH-SET este gol, adică KSTATE0/4
                                       păstrează +FF.
                INC   HL               Totuși, dacă setul nu este
                DEC   (HL)             gol, decrementează al său
                DEC   HL               'numărător de 5 cereri'
                JR    NZ,02D1,K-CH-SET si cînd ajunge zero,
                LD    (HL),+FF         se semnalizează setul
                                       ca fiind gol.
```

După ce s-a luat în considerare primul set, se schimbă indicatorul si se ia în considerare al 2-lea set.

```
02D1 K-CH-SET   LD    A,L              Scoate bitul cel mai putin
                LD    HL,+KSTATE4      semnificativ al adresei si
                CP    L                salt înapoi dacă al 2-lea
                JR    NZ,02C6,K-ST-LOOP set este încă luat în
                                       considerare.
```

Întoarcere acum, dacă valoarea tastei indică 'nici o tastă' sau numai o tastă shift.

```
                CALL  031E,K-TEST      Se fac testele necesare si
                RET   NC               si întoarcere dacă este
                                       nevoie. Se schimbă de
                                       asemenea valoarea tastei cu
                                       un 'cod principal'.
```

O lovitură de tastă ce se repetă (menținută apăsată) este separată de o nouă lovitură de tastă.

```
                LD    HL,+KSTATE0      Priveste întîi la KSTATE0.
                CP    (HL)             Salt mai departe dacă
                JR    Z,0310,K-REPEAT  codurile care se întrec
                                       indică o repetare.
                EX    DE,HL            Salvează adresa lui
                                       KSTATE0.
                LD    HL,+KSTATE4      Acum se verifică KSTATE4.
                CP    (HL)             Salt mai departe dacă
                JR    Z,0310,K-REPEAT  codurile ce se întrec indică
```

o repetare.

Dar o nouă tastă nu va fi acceptată decît dacă unul din seturile de variabile sistem KSTATE este 'gol'.

```
        BIT   7,(HL)          Se consideră al 2-lea set.
        JR    NZ,02F1,K-NEW   Salt mai departe dacă este
                              'gol'.
        EX    DE,HL           Acum se consideră primul set.
        BIT   7,(HL)          Continuă dacă setul este
        RET   Z               'gol', dacă nu părăseste
                              subrutina KEYBOARD
                              (tastatură).
```

Noua tastă este acceptată. Inainte ca variabila sistem LAST-K să poată fi umplută, variabilele sistem KSTATE, ale setului folosit, trebuie să fie initializate pentru a trata orice repetări, iar codul tastei trebuie decodat.

```
02F1 K-NEW    LD    E,A       Codul este pasat la registrul
              LD    (HL),A    E si la KSTATE0/4.
              INC   HL        'Numărătorul de 5 cereri'
              LD    (HL),+05  pentru acest set este resetat
                              la '5'.
              INC   HL        A treia variabilă sistem a
              LD    A,(REPDEL) setului mentine valoarea
              LD    (HL),A    REPDEL (normal 0,7 sec).
              INC   HL        Indică la KSTATE3/7.
```

Decodarea unui 'cod principal' depinde de starea prezentă a lui MODE, bitul 3 de la FLAGS si de 'bitul de shift'.

```
              LD    C,(MODE)  Scoate MODE.
              LD    B,(FLAGS) Scoate FLAGS.
              PUSH  HL        Salvează pointerul în timp ce
              CALL  0333,K-DECODE 'codul principal' este
              POP   HL        decodat.
              LD    (HL),A    Valoarea finală a codului
                              este salvată în KSTATE3/7,
                              de unde el este cules în
                              cazul unei repetări.
```

Următoarele trei linii de instructii sînt comune pentru tratarea atît a 'tastelor noi' cît si a 'tastelor repetabile'.

```
0308 K-END    LD    (LAST-K),A Se introduce valoarea
              SET   5,(FLAG)  finală a codului în LAST-K.
              RET             Intoarcere finală.
```

THE 'REPEATING KEY' SUBROUTINE (REPETARE TASTA)
O tastă se va 'repeta' într-o primă fază după perioada de stergere - REPDEL (normal 0,7 sec.) iar în fazele următoare după perioada de stergere - REPER (normal 0,1 sec.).

```
0310 K-REPEAT INC   HL        Se indică la numărătorul de
              LD    (HL),+05  '5 cereri' setul care se va
                              folosi si se resetează la
                              '5'.
              INC   HL        Se indică la a 3-a variabilă
              DEC   (HL)      de sistem valoarea
                              REPDEL/REPPER si se
                              decrementează.
              RET   NZ        Se iasă din rutina
                              KEYBOARD (tastatură) dacă
                              nu a trecut perioada de
                              stergere.
              LD    A,(REPPER) Totusi, o dată ce a trecut
              LD    (HL),A    perioada de stergere, pentru
                              următoarea repetare este
                              REPPER.
              INC   HL        Repetarea a fost acceptată
              LD    A,(HL)    astfel că valoarea finală a
              JR    0308,K-END codului este scoasă de
                              KSTATE3/7 si pasată la K-END.
```

THE 'K-TEST' SUBROUTINE
Se testează valoarea tastei si se face o întoarcere dacă este 'nici o tastă' sau 'numai shift'; în caz contrar este găsit 'codul principal' al acestei taste.

```
031E K-TEST     LD    E,D          Copiază bitul shift.
                LD    D,+00        Sterge registrul D pentru
                                   mai tîrziu.
                LD    A,E          Transferă numărul tastei.
                CP    +27          Acum se execută întoarcere
                RET   NC           dacă tasta a fost numai 'CAPS
                                   SHIFT' sau 'nici o tastă'.
                CP    +18          Salt mai departe în afară
                JR    NZ,032C,K-MAIN de cazul că tasta 'E' a fost
                                   SYMBOL SHIFT si o altă
                                   tastă.
                BIT   7,B          Totusi se acceptă SYMBOL
                RET   NZ           SHIFT si o altă tastă;
                                   întoarcere numai cu SYMBOL
                                   SHIFT.
```

'Codul principal' este găsit prin indexarea tabelului principal de taste.

```
032C K-MAIN     LD    HL,0205      Adresa de bază a tabelului.
                ADD   HL,DE        Se indexează în tabel si se
                LD    A,(HL)       aduce 'codul principal'.
                SCF                Semnalizează 'lovitură tastă
                RET                validă' înainte de
                                   întoarcere.
```

THE KEYBOARD DECODING' SUBROUTINE (DECODARE TASTATURA)
Această subrutină este începută cu 'codul principal' în registrul E, valoarea lui FLAGS în registrul D, valoarea lui MODE în registrul C si 'bitul shift' în registrul B.
Luînd în considerare aceste patru valori si referindu-ne, dacă este necesar, la cele 6 tabele de taste, se va produce 'codul final'. Acesta este întors în registrul A.

```
0333 K-DECODE   LD    A,E          Copiază 'codul principal'.
                CP    +3A          Salt mai departe dacă este
                JR    C,0367,K-DIGIT considerată o tastă
                                   digitală; de asemenea SPACE,
                                   ENTER si ambele shiftate.
                DEC   C            Decrementează valoarea MODE.
                JP    M,034F,K-KLC-LET Salt mai departe pentru
                JR    Z,0341,K-E-LET modurile 'K', 'L', 'C' & 'E'.
```

Rămîne doar modul 'grafic' iar 'codul final' pentru tastele literale în modul grafic este calculat din 'codul principal'.

```
                ADD   A,+4F        Adună offset-ul.
                RET                Întoarcere cu 'codul final'.
```

In continuare sînt luate în considerare tastele literale în modul extins.

```
0341 K-E-LET    LD    HL,+01E8     Adresa de bază pentru tabelul
                                   'b'.
                INC   B            Dacă nici o tastă shift nu a
                JR    Z,034A,K-LOOK-UP fost apăsată, salt mai
                                   departe pentru a folosi
                                   acest tabel.
                LD    HL,0205      În caz contrar se foloseste
                                   adresa de bază pentru tabelul
                                   'c'.
```

Tabelele de taste 'b-f' sînt toate aservite de următoarea rutină de urmărire. În toate cazurile este servit si întors un 'cod final'.

```
034A L-LOOK-UP  LD    D,+00        Se sterge registrul D.
                ADD   HL,DE        Se indexează tabelul cerut si
                LD    A,(HL)       se aduce 'codul final'.
                RET                Apoi întoarcere.
```

Se vor considera acum tastele literale în modurile 'K', 'L' sau 'C'. Dar mai întîi trebuie tratate cu codurile speciale SYMBOL SHIFT.

```
034F K-KLC-LET  LD    HL,+0229     Adresa de bază pentru tabelul
                                   'e'.
                BIT   0,B          Salt înapoi dacă se
                JR    Z,034A,K-LOOK-UP foloseste tasta SYMBOL SHIFT
                                   si o tastă literală.
                BIT   3,D          Salt mai departe dacă esti
                JR    Z,0364,K-TOKENS curent în modul 'K'.
                BIT   3,(FLAGS2)   Dacă este fixat CAPS LOCK,
```

```
                RET   NZ                atunci întoarcere cu 'codul
                                        principal'.
                INC   B                 Întoarcere în aceeași manieră
                RET   NZ                dacă CAPS SHIFT a fost
                                        apăsat.
                ADD   A,+20             Dacă sînt cerute coduri mici
                RET                     atunci trebuie să se adune
                                        +20 la 'codul principal'
                                        pentru a da 'codul final'
                                        corect.
```

Valorile de 'cod final' pentru simboluri sînt găsite adăugîndu +A5 la 'codul principal'.

```
0364 K-TOKENS   ADD   A,+A5             Se adună offset-ul cerut și
                RET                     întoarcere.
```

În continuare se consideră tastele digitale și SPACE, ENTER și amîndouă shiftate.

```
0367 K-DIGIT    CP    +30               Continuă numai cu tastele
                                        digitale,
                RET   C                 adică întoarcere cu SPACE
                                        (+20), ENTER (+0D) și
                                        amîndouă siftate (+0E).
                DEC   C                 Acum separă tastele digitale
                                        în trei grupe, în
                                        corespondență cu modul de
                                        lucru.
                JP    M,039D,K-KLC-DGT  Sari cu modurile 'K', 'L' și
                                        'C'
                JR    NZ,0389,K-GRA-DGT si cu modul 'G' de asemenea.
                                        Continuă cu modul 'E'.
                LD    HL,+0254          Adresa de bază a tabelului
                                        'f'.
                BIT   5,B               Folosește acest tabel pentru
                JR    Z,034A,K-LOOK-UP  SYMBOL SHIFT și o tastă
                                        digitală în modul extins.
                CP    +38               Sari mai departe cu tastele
                JR    NC,0382,K-8-&-9   digitale '8' și '9'.
```

Tastele digitale de la '0' la '7' în modul extins sînt pentru a da ori un 'cod culoare hîrtie' ori un 'cod culoare cernealâ', depinzînd de modul de folosire a lui CAPS SHIFT.

```
                SUB   +20               Se reduce domeniul de la +30
                                        la +37, oferind un domeniu
                                        de la +10 la +17.
                INC   B                 Întoarcere cu acest 'cod de
                RET   Z                 culoare hîrtie' dacă CAPS
                                        SHIFT nu a fost folosit.
                ADD   A,+08             Dar dacă el este, domeniul
                RET                     în schimb este de la +18
                                        la +1f - indicînd un 'cod
                                        culoare cernealâ'.
```

Tastele digitale '8' și '9' sînt pentru a da codurile 'BRIGHT' (LUMINOS) și respectiv 'FLASH'.

```
0382 K-8-&-9    SUB   +36               +38 și +39 ajung +02 și
                                        respectiv +03.
                INC   B                 Întoarcere cu aceste coduri
                RET   Z                 dacă CAPS SHIFT nu a fost
                                        folosit. (Acestea sînt
                                        codurile (BRIGHT').
                ADD   A,+FE             Scădere '2' dacă CAPS SHIFT
                RET                     este folosit, obținînd +00
                                        și respectiv +01 (ca și
                                        coduri 'FLASH').
```

Tastele digitale în modul grafic sînt pentru a furniza blocul de caractere grafice (+80 ÷ +8F), codul GRAFICS (+0F) și codul DELETE (+0C) - STERGERE.

```
0389 K-GRA-DGT  LD    HL,+0230          Adresa de bază a tabelului
                                        'd'.
                CP    +39               Folosește acest tabel
                JR    Z,034A,K-LOOK-UP  direct pentru ambele taste
                CP    +30               digitale '9' pentru a
                JP    Z,034A,K-LOOK-UP  GRAFICS și tastă digitală
```

```
                                                  '0'   pentru   a   furniza
                                                  DELETE.
                AND   +07                         Pentru  tastele  de la '1' la
                ADD   A,+80                       '8' se face domeniul +80 +87.
                INC   B                           Intoarcere cu o valoare din
                RET   Z                           acest domeniu  dacă nici una
                                                  din tastele SHIFT nu a fost
                                                  apăsată.
                XOR   +0F                         Dar  dacă  s-a  shiftat,  fă
                RET                               domeniul între +88 si +8F.
```

In final, consideràm tastele digitale în modurile 'K', 'L' si respectiv 'C'.

```
039D K-KLC-DGT  INC   B                           Intoarcere directă dacă nu
                RET   Z                           s-a  folosit  nici  o  tastă
                                                  SHIFT (codurile  finale  de
                                                  la +30 la +39).
                BIT   5,B                         Foloseste  tabelul  'd'  dacă
                LD    HL,+0230                    tasta  CAPS SHIFT  a  fost de
                JR    NZ,034A,K-LOOK-UP           asemenea apăsată.
```

Acum pot fi găsite codurile pentru diferite taste digitale si SYMBOL SHIFT.

```
                SUB   +10                         Redu  domeniul   pentru   a
                                                  furniza +20 ÷ +29.
                CP    +22                         Separă    caracterul   '@'   de
                JR    Z,03B2,K-@-CHAR             celelalte.
                CP    +20                         Caracterul  '-'  trebuie  de
                                                  asemenea separat.
                RET   NZ                          Acum  întoarcere cu 'codurile
                                                  finale' +21, +23 ÷ +29.
                LD    A,+5F                       Dă caracterului '-' un cod de
                RET                               +5F.
03B2 K-@-CHAR   LD    A,+40                       Dă caracterului '@' un cod de
                RET                               +40.
```

RUTINELE DE DIFUZOR

Cele două subrutine care se prezintă în această secțiune sînt subrutina BEEPER, care controlează difuzorul, și rutina de comandă BEEP.
Difuzorul este activat prin D4 pe nivel '0' logic în timpul unei instrucții OUT folosind portul '254'. Intr-o situație similară, cînd D4 este pe nivel '1' logic, difuzorul este dezactivat. De aceea un 'beep' se va produce prin modificarea regulată a nivelului lui D4.

Acum 'nota de mijloc C' care are frecvența 261,63 Hz. Pentru a produce această notă, difuzorul va fi activat și dezactivat alternativ la fiecare a 1/523,26-a parte dintr-o secundă. In SPECTRUM sistemul clock funcționează la 3,5 MHz, iar 'nota de mijloc C' va necesita ca instrucțiia OUT necesară să fie executată pe cît posibil la fiecare din stările 6,689 T. Această ultimă valoare, redusă ușor la depășiri inevitabile, reprezintă în subrutina BEEPER 'lungimea buclei de timp'.

THE 'BEEPER' SUBROUTINE (SUBRUTINA BEEPER)
Subrutina începe cu registrul pereche DE conținînd valoarea 'f*t', unde o notă de frecvență 'f' dată va avea o durată de 't' secunde și cu registrul pereche HL conținînd o valoare egală cu numărul de stări T din 'bucla de timp' împărțit cu '4'.
Adică, pentru ca să fie produsă 'nota de mijloc C', DE va conține pentru o secundă +0103 (INT (261.63*1)) și HL va conține +066A (derivat de la 6.689/4 - 30.125).

```
03B5 BEEPER     DI                          Dezactivează întreruperea pe
                                            durata unui 'beep'.
                LD    A,L                   Salvează temporar L.
                SRL   L                     Fiecare '1' din registrul L
                SRL   L                     este pentru a număra '4'
                                            stări T dar, în schimb
                                            acceptă INT (L/4) și numără
                                            '16' stări T.
                CPL                         Intoarcere la valoarea
                AND   +03                   originală din L și află cîți
                LD    C,A                   de '1' s-au pierdut prin
                LD    B,+00                 acceptarea lui INT (L/4).
                LD    IX,+03D1              Adresa de bază a buclei de
                                            timp.
                ADD   IX,BC                 Modifică lungimea buclei de
                                            timp. Folosește un punct
                                            de pornire prim pentru
                                            fiecare '1' pierdut prin
                                            acceptarea lui INT (L/4).
                LD    A,(BORDCR)            Scoate culoarea de margine
                AND   +39                   prezentă și mut-o la biții 2,
                RRCA                        1 și 0 ai registrului A.
                RRCA
                OR    +08                   Asigură-te că ieșirea MIC
                                            este 'off'.
```

Acum se intră în bucla de generare a sunetului. Se fac treceri complete prin 'DE', adică un pas pentru fiecare ciclu al notei.
Registrul HL păstrează lungimea buclei de timp pentru fiecare '1' din registrul L fiind folosite '16' stări T și '1,024' stări T pentru fiecare '1' din registrul H.

```
03D1 BE-IX+3    NOP                         Adună '4' stări T pentru
03D2 BE-IX+2    NOP                         fiecare punct prim de intrare
03D3 BE-IX+1    NOP                         folosit.
03D4 BE-IX+0    INC   B                     Valorile din registrele B&C
                INC   C                     vin din registrele H&L - după
                                            cum se vede mai jos.
03D6 BE-H&L-LP  DEC   C                     Bucla de timp.
                JR    NZ,03D6,BE-H&L-LP     Adică 'BC' * '4' stări T.
                LD    C,+3F                 (Dar notează că în punctul de
                DEC   B                     jumătate ciclu - C va fi egal
                JP    NZ,03D6,BE-H&L-LP     cu 'L+1').
```

In continuare difuzorul va fi activat și dezactivat alternativ.

```
                XOR   +10                   Basculează bitul 4.
                OUT   (+FE),A               Execută operația OUT lăsînd
                                            marginea neschimbată.
                LD    B,H                   Resetează registrul B.
                LD    C,A                   Salvează registrul A.
                BIT   4,A                   Salt dacă ești la punctul ce
```

```
                JR    NZ,03F2,BE-AGAIN        indică jumătate de ciclu.
```

După un ciclu complet este testat registrul pereche BE.

```
                LD    A,B                     Salt în față dacă ultimul pas
                OR    E                       complet a fost deja făcut.
                JR    I,03F6,BE-END
                LD    A,C                     Se aduce valoarea salvată.
                LD    C,L                     Resetare registrul C.
                DEC   BE                      Se decrementează numărătorul
                                              de pași.
                JP    (IX)                    Salt înapoi la locația de
                                              început curută a buclei.
```

Sînt stabiliți parametrii pentru a doua jumătate de ciclu.

```
03F2 BE-AGAIN   LD    C,L                     Resetare registru C.
                INC   C                       Cum acest pas este mai scurt,
                                              se adună '16' stări T.
                JP    (IX)                    Salt înapoi.
```

Cu privire la completarea 'beep'-ului, întreruperea mascabilă trebuie să fie validată.

```
03F6 BE-END     EI                            Întrerupere validată.
                RET                           Întrerupere finală.
```

THE 'BEEP' COMMAND ROUTINE (RUTINA DE COMANDA BEEP)
Această subrutină începe cu două numere în stiva calculatorului. Numărul cel mai de sus reprezintă 'nota', iar numărul imediat sub el reprezintă 'durata'.

```
03F8 BEEP       RST   0028,FP-CALC            Virgula mobilă este folosită
                                              pentru a manipula cele două
                                              valori - t & P.
                DEFB  +31,duplicate           t, P, P
                DEFB  +27,int                 t, P, i (unde i = INT P)
                DEFB  +C0,st-mem-0            t, P, i (mem-0 păstrează i)
                DEFB  +03,subtract            t, p (unde p este partea
                                              fracțională a lui P)
                DEFB  +34,stk-data            Stochează valoarea zecimală a
                                              lui 'K'.
                DEFB  +EC,exponent+7C         0.0577622606 (care este putin
                DEFB  +6C,+98,+1F,+F5         sub 12 V2 - 1)
                DEFB  +04,multiply            t, pK
                DEFB  +A1,stk-one             t, pK, 1
                DEFB  +0F,addition            t, pK+1
                DEFB  +38,end-calc
```

Acum se vor efectua mai multe teste asupra lui i, partea 'întreagă a intensității'.

```
                LD    HL,+5C92                Aceasta este primul 'mem-0'
                                              (MEMBOT).
                LD    A,(HL)                  Adu exponentul lui i.
                AND   A                       Dacă i nu este în forma
                JR    NZ,046C,REPORT-B        integrală (scurtă) atunci se
                                              dă o eroare.
                INC   HL                      Copiază bitul semn în
                LD    C,(HL)                  registrul C.
                INC   HL                      Copiază bitul cel mai putin
                LD    B,(HL)                  semnificativ în registrul B
                LD    A,B                     și în registrul A.
                RLA                           Se dă iar un report B dacă i
                SBC   A,A                     nu satisface testul:
                CP    C                       -128<=i<=+127
                JR    NZ,046C,REPORT-B
                INC   HL
                CP    (HL)
                JR    NZ,046C,REPORT-B
                LD    A,B                     Adu bitul mai putin
                                              semnificativ și testează-l
                                              mai departe.
                ADD   A,+3C
                JP    P,0425,BE-i-OK          Acceptă -60<=i<=67
                JP    PO,046C,REPORT-B        Respinge de la -128 la -61.
```

Notă: Domeniul de la +70 la +127 se va respinge mai tîrziu.

Acum se va putea găsi frecventa corectă pentru 'intensitatea' i.

```
0425 BE-i-OK      LD    B,+FA                 Se începe cu 6 octave sub
                                              nota de mijloc C.
0427 BE-OCTAVE    INC   B                     Pentru a afla octava corectă
                  SUB   +0C                   se va reduce regulat i.
                  JR    NC,0427,BE-OCTAVE
                  ADD   A,+0C                 Adună înapoi ultima scădere
                                              făcută.
                  PUSH  BC                    Salvează numărul octavei.
                  LD    HL,+046E              Adresa de bază a 'tabelei de
                                              semitonuri'.
                  CALL  3406,LOC-MEM          Se ia în considerare tabelul
                  CALL  33B4,STACK-NUM        și se merge la stiva
                                              calculatorului la a 'A'-a
                                              valoare. (Numește-o C).
```

Acum se poate lua în considerare partea fractională a 'intensității' i.

```
                  RST   0028,FP-CALC          t, pK+i, C
                  DEFB  +04,multiply          t, C(pK+i)
                  DEFB  +38,end-calc
```

Prin modificarea 'ultimei valori', în concordanță cu numărul octavei, se găsește frecvența finală f.

```
                  POP   AF                    Se scoate numărul octavei.
                  ADD   A,(HL)                Se multiplică 'ultima
                  LD    (HL),A                valoare' cu '2 la puterea
                                              numărului octavei'.
                  RST   0028,FP-CALC          t, f
                  DEFB  +C0,st-mem-0          Pentru moment frecvența se
                  DEFB  +02,delete            pune deoparte în mem-0.
```

Atenție în noua turnură la 'durată'.

```
                  DEFB  +31,duplicate         t, t
                  DEFB  +38,end-calc
                  CALL  1E94,FIND-INT1        Noua valoare 'INT t' trebuie
                  CP    +0B                   să fie în domwniul +00 - +0A.
                  JR    NC,046C,REPORT-B
```

Numărul complet de cicluri din 'beep' este dat de 'f*t'. Astfel acum se poate găsi noua valoare.

```
                  RST   0028,FP-CALC          t
                  DEFB  +E0,get-mem-0         t, f
                  DEFB  +04,multiply          f*t
```

Rezultatul este lăsat în stiva calculatorului în timp ce este calculată lungimea 'buclei de timp' cerută de 'beep'.

```
                  DEFB  +E0,get-mem-0         f*t, f
                  DEFB  +34,stk-data          Valoarea '3.5*10 6/8' este
                  DEFB  +80,four bytes        formată în vîrful stivei
                  DEFB  +43,exponent +93      calculatorului.
                  DEFB  +55,+9F,+80,(+00)     f*t, f, 437,500 (dec.)
                  DEFB  +01,exchange          f*t, 437,500, f
                  DEFB  +05,division          f*t, 437,500/f
                  DEFB  +34,stk-data
                  DEFB  +35,exponent +85
                  DEFB  +71,(+00,+00,+00)     f*t, 437,500/f, 30,125 (dec)
                  DEFB  +03,subtract          f*t, 437,500/f - 30,125
                  DEFB  +38,end-calc
```

Notă: Valoarea '437,500/f' dă lungimea 'jumătății de ciclu' a notei și reducînd-o cu '30,125' se iau în considerare stările T '120,5' în care se produce nota și se ajustează contoarele etc.
Acum se vor putea transfera valorile în registrele cerute.

```
                  CALL  1E99,FIND-INT2        Valoarea 'buclei de timp'
                  PUSH  BC                    este în registrul pereche BC;
                                              este salvată în stivă.
```

Notă: Dacă bucla de timp este prea mare, atunci va apare o eroare (reîntorcîndu-se prin ERROR-1); de aceea se exclud valorile 'intensității' din domeniul '+70 - +127'.

```
                  CALL  1E99,FIND-INT2        Valoarea f*t este comprimată
                                              în registrul pereche BC.
```

```
                POP   HL                Se mută valoarea 'buclei de
                                        timp' în registrul pereche
                                        HL.
                LD    B,B               Se mută valoarea făt în
                LD    E,C               registrul pereche DE.
```

Totuși, înainte de a face 'beep'-ul, se testează valoarea făt.

```
                LD    A,E               Dacă făt a dat rezultatul că
                OR    E                 nu e necesar nici un ciclu,
                RET   Z                 atunci reîntoarcere.
                DEC   DE                Se descreste numărul de ciclu
                JP    0385,BEEPER       si se sare la rutina BEEPER
                                        (făcîndu-se în sfîrsit un
                                        pas).
```

Prezentarea B - INTEGER în afara domeniului

```
046C REPORT-B   RST   0008,ERROR-1       Cheamă rutina de tratare a
                DEFB  +0A                erorii.
```

THE 'SEMI-TONE' TABLE (TABELUL DE 'SEMI-TONE')
Acest tabel contine frecventele a 12 semitonuri dintr-o octavă.

```
                                          frecventa Hz          nota
046E DEFB +89,+02,+D0,+12,+86             261.63                C
     DEFB +89,+0a,+97,+60,+75             277.18                C#
     DEFB +89,+12,+B5,+17,+1F             293.66                D
     DEFB +89,+1B,+90,+41,+02             311.13                D#
     DEFB +89,+24,+D0,+53,+CA             329.63                E
     DEFB +89,+2E,+9D,+36,+E1             349.23                F
     DEFB +89,+38,+FF,+49,+3E             369.99                F#
     DEFB +89,+43,+FF,+6A,+73             392                   G
     DEFB +89,+4F,+A7,+00,+54             415.30                G#
     DEFB +89,+5C,+00,+00,+00             440                   A
     DEFB +89,+69,+14,+F6,+24             466.16                A#
     DEFB +89,+76,+F1,+10,+05             493.88                B
```

THE 'PROGRAM NAME' SUBROUTINE (ZX81)
Subrutina următoare se aplică la ZX81 si nu a fost scoasă când programul a fost rescris pentru SPECTRUM.

```
04AA DEFB +CB,+F8,+24,+3A
     DEFB +38,+5C,+87,+FA
     DEFB +8A,+1C,+E1,+D0
     DEFB +E5,+CD,+F1,+2B
     DEFB +62,+6B,+09,+F8
     DEFB +09,+CB,+FE,+C9
```

THE CASSETTE HANDLING ROUTINES (RUTINELE DE TRATARE A CASETEI)

Programul monitor de 16K are un set extins de rutine pentru tratarea interfetei cu caseta. In consecintă, aceste rutine formează rutinele de comanda SAVE, LOAD, VERIFY si MERGE.
Punctul de intrare al rutinelor este la SAVE-ETC (0605). Cu toate acestea, înaintea acestui punct se găsesc subrutinele referitoare la salvarea si încărcarea (sau verificarea) bitilor.
In toate cazurile, bitii ce se vor trata cu aceste subrutine sînt descrisi de registrul pereche DE ce contine 'lungimea blocului', de registrul pereche IX ce contine 'adresa de bază' si de registrul A ce contine +00 pentru blocul de vîrf sau +FF pentru un bloc program/data.

THE 'SA-BYTES' SUBROUTINE (SUBRUTINA 'SA-BYTES')
Această subrutină este chemată să salveze informatia de vîrf (de la 098A) si mai tîrziu blocul actual program/data (de la 099E).

```
04C2 SA-BYTES   LD    HL,+053F        Preîncarcă stiva masinii cu
                PUSH  HL              adresa - SA/LD-RET.
                LD    HL,+1F80        Această constantă va da un
                                      rol conducător unei
                                      informatii de vîrf pentru
                                      aprox.5 sec.
                BIT   7,A             Sari mai departe dacă se
                JR    Z,04D0,SA-FLAG  salvează un header.
                LD    HL,+0C98        Această constantă va da un
                                      rol conducător blocului
                                      program/data pentru aprox.2
                                      secunde.
04D0 SA-FLAG    EX    AF,A'F'         Indicatorul este salvat.
                INC   DE              'Lungimea' este incrementată
                DEC   IX              si 'adresa de bază' este
                                      redusă tinînd seama de
                                      indicator.
                DI                    Intreruperea mascabilă este
                                      dezactivată în timpul lui
                                      SAVE.
                LD    A,+02           Semnalul 'MIC on' si marginea
                                      să fie RED.
                LD    B,A             Se dă o valoare lui B.
```

Se intră acum într-o buclă pentru a crea impulsurile conducătorului. Atît impulsurile 'MIC on' cît si cele 'MIC off' sînt de lungime 2,168 stări T. Culoarea de margine se va schimba pe fiecare 'front' de la RED la CYAN.

Notă: Se consideră un 'front' ca fiind o tranzitie atît de la 'on' la 'off', cît si de la 'off' la 'on'.

```
04D8 SA-LEADER  DJNZ  04D8,SA-LEADER  Perioada de timp principală.
                OUT   (+FE),A         La fiecare pas, MIC on/off,
                XOR   +0F             margine RED/CYAN.
                LD    B,+A4           Constanta de timp principală.
                DEC   L               Se decrementează numărătorul
                                      cel mai putin semnificativ.
                JR    NZ,04D8,SA-LEADER  Salt înapoi pentru un alt
                                      impuls.
                DEC   B               Se tine seamă de un pas mai
                                      lung (-redus cu 13 stări T).
                DEC   H               Se decrementează numărătorul
                                      cel mai semnificativ.
                JP    P,04D8,SA-LEADER  Salt înapoi pentru alt impuls
                                      pînă la stabilirea
                                      conducătorului.
```

Se trimite acum un impuls de sincronizare.

```
                LD    B,+2F
04EA SA-SYNC-1  DJNZ  04EA,SA-SYNC-1  MIC off pentru 667 stări T de
                                      la 'OUT to OUT'.
                OUT   (+FE),A         MIC on si RED.
                LD    A,0B            Semnal 'MIC off & CYAN'.
                LD    B,+37           MIC on pentru 735 stări de la
04F2 SA-SYNC-2  DJNZ  04F2,SA-SYNC-2  'OUT to OUT'.
                OUT   (+FE),A         Acum MIC off & margine CYAN.
                LD    BC,+3B0E        +3B este o constantă de timp;
                                      +0E semnal 'MIC off & YELLOW'
                EX    AF,A'F'         Incarcă fanionul si îl treci
                LD    L,A             în registrul L pentru
```

```
                        JP    0507,SA-START      'trimitere'.
                                                 Intoarcere în bucla SAVE
                                                 (SALVARE).
```

Octetul SAVE (SALVARE) este acum introdus. Primul octet ce urmează a fi salvat este indicatorul; acesta este urmat de octetul informatie actual, iar ultimul octet construit este octetul paritate, care se formează prin evaluarea bitilor de mai înainte.

```
04FE SA-LOOP        LD    A,D          Numărătorul lungime este
                    OR    E            testat si saltul se face dacă
                    JR    Z,050E,SA-PARITY  a ajuns pe zero.
                    LD    L,(IX+00)    Incarcă următorul octet care
                                       urmează să fie salvat.
0505 SA-LOOP-P      LD    A,H          Incarcă 'paritatea' curentă.
                    XOR   L            Include octetul actual.
0507 SA-START       LD    H,A          Rememorează 'paritatea'.
                                       Atentie: pe această ramură
                                       valoarea indicatorului
                                       intializează 'paritatea'.
                    LD    A,+01        Semnal 'MIC on & BLUE'.
                    SCF                Setează fanionul de transport
                                       Va functiona ca un 'marcator'
                                       pentru cei 8 biti ai
                                       octetului.
                    JP    0525,SA-8-BITES  Salt înainte.
```

Cînd trebuie trimis octetul 'paritate', atunci el este transferat în registrul L pentru a fi salvat.

```
050E SA-PARITY      LD    L,H          Ia valoarea 'paritate finală'
                    JR    0505,SA-LOOP-P  Salt înapoi.
```

Următoarea buclă internă produce actualul tact. Accesul în buclă se face la SA-BIT-1 cu tipul bitului de salvat indicat de fanionul de transport. Pentru fiecare bit se fac două treceri prin buclă, realizînd astfel un 'tact off' si nu un 'tact on'.

```
0511 SA-BIT-2       LD    A,C          Adus la a doua trecere si se
                                       încarcă 'MIC off & YELLOW'.
                    BIT   7,B          Pune fanionul zero să indice
                                       'a doua trecere'.
0514 SA-BIT-1       DJNZ  0514,SA-BIT-1  Aceeasi buclă de sincronizare
                                       Totdeauna 801 stări T la a
                                       doua trecere.
                    JR    NC,051C,SA-OUT  Salt, pe calea cea mai scurtă
                                       dacă se salvează '0'.
                    LD    B,+42        Oricum, dacă se salvează '1',
051A SA-SET         DJNZ  051A,SA-SET  atunci se adună 855 stări T:
051C SA-OUT         OUT   (+FE),A      La prima trecere este 'MIC on
                                       & BLUE' iar la a doua trecere
                                       este 'MIC off & YELLOW'.
                    LD    B,+3E        Incarcă constanta de
                                       sincronizare pentru a doua
                                       trecere.
                    JR    NZ,0511,SA-BIT-2  Salt înapoi la sfîrsitul
                    DEC   B            primei treceri; altfel se
                                       refac 13 stări T.
                    XOR   A            Sterge fanionul de transport
                    INC   A            si pune în A +01 (MIC on &
                                       BLUE) înainte de a continua
                                       'bucla celor 8 biti'.
```

'Bucla celor 8 biti' este accesată initial cu întregul octet în registrul L si indicatorul de transport este 1. Oricum, este reentrantă după fiecare bit ce a fost salvat pînă cînd s-a ajuns la capăt, cînd 'marcatorul' trece în indicatorul de transport, lăsînd registrul L plin.

```
0525 SA-8-BITS      RL    L            Pune bitul 7 în indicatorul
                                       de transport si deplasează la
                                       stînga 'marcatorul'.
                    JP    NZ,0514,SA-BIT-1  Salvare bit pînă la terminare
                                       octet.
                    DEC   D,E          Decrementare 'contor'.
                    INC   IX           Avans 'adresa bază'.
                    LD    B,+31        Setare constantă de
                                       sincronizare pentru primul
                                       bit al următorului octet.
                    LD    A,+7F        Intoarcere (la SA/LD-RET)
```

```
                IN    A,(+FE)          dacă tasta BREAK (PAUZA) a
                RRA                    fost apăsată.
                RET   NC
                LD    A,D              Testare 'contor' si salt
                INC   A                înapoi chiar dacă este zero
                JP    NZ,04FE,SA-LOOP  (pentru a trimite octetul
                                       'paritate').
                LD    B,+3B            Iesire cînd 'contorul' a
053C SA-DELAY   DJNZ  053C,SA-DELAY    ajuns la +FFFF. Dar înainte
                RET                    se dă o scurtă întîrziere.
```

Notă: O resetare va da pentru 'MIC off' tact pentru 855 stări T urmate de un 'MIC on' tact pentru 855 stări T. Se ia în considerare că setarea unui bit va da tacturi exact de două ori mai lungi. De asemenea nu există intervale nici între tactul sincronizare si primul bit al indicatorului, nici între octeti.

THE 'SA/LD-RET' SUBROUTINE
Această subrutină este comună pentru SALVAre si INCARCAre.
Marginea ecranului este pusă pe culoarea originală, iar tasta BREAK testată pentru ultima dată.

```
053F SA/LD-RET  PUSH  AF               Salvare indicator de
                                       transport. (Este resetat după
                                       o încărcare eronată).
                LD    A,(BORDCR)       Se aduce culoarea originală
                AND   +38              a marginii din variabila
                                       sa sistem.
                RRCA                   Mută culoarea marginii în
                RRCA                   bitii 2, 1 si 0.
                RRCA
                OUT   (+FE),A          Pune marginea pe culoarea
                                       originară.
                LD    A,+7F            Citeste tasta BREAK pentru
                IN    A,(+FE)          ultima oară.
                RRA
                EI                     Autorizare întrerupere
                                       mascabilă.
                JR    C,0554,SA/LD-END Salt numai dacă urmează o
                                       pauză.
```

Prezentarea D-BREAK-CONT se repetă.

```
0552 REPORT-D   RST   0008,ERROR-1     Apel rutină tratare eroare.
                DEFB  +0C
```

Se continuă aici.

```
0554 SA/LD-END  POP   AF               Regăsire fanion de transport.
                RET                    Intoarcere în rutina
                                       apelantă.
```

THE 'LD-BYTES' SUBROUTINE
Această subrutină este apelată pentru încărcarea (LOAD) header-ului informatiei (de la 076E) si încărcarea întîrziată (LOAD) sau verificarea (VERIFY) unui set actual de informatii (de la 0802).

```
0556 LD-BYTES   INC   D                Resetează fanionul zero (în D
                                       nu poate fi +FF).
                EX    AF,A'F'          Registrul A contine +00
                                       pentru header si +FF pentru
                                       un set de informatii.
                                       Fanionul de transport este
                                       resetat pentru verificare
                                       (VERIFY) si setat pentru
                                       încărcare (LOAD).
                DEC   D                Refă D la valoarea de la care
                                       a plecat.
                DI                     Intreruperea mascabilă este
                                       acum dezautorizată.
                LD    A,+0F            Marginea se face acum WHITE
                                       (albă).
                OUT   (+FE),A
                LD    HL,+053F         Incarcă prima dată în stiva
                PUSH  HL               calculatorului adresa
                                       SA/LD-RET.
                IN    A,(+FE)          Se face o citire initială a
                                       portului '254'.
                RRA                    Se roteste octetul obtinut,
                AND   +20              dar se păstrează numai bitul
```

```
                                                    EAR.
                OR      +02                         Semnal margine 'RED' (rosu).
                LD      C,A                         Se păstrează valoarea în
                                                    registrul C (+22 pentru 'off'
                                                    si +02 pentru 'on' - starea
                                                    prezentă pentru EAR).
                CP      A                           Se setează fanionul zero.
```

Primul stadiu al citirii benzii implică demonstrarea faptului că semnalul pulsat actual există (de exemplu, tranzitiile 'on/off' sau 'off/on').

```
056B LD-BREAK   RET     NZ                          Revenire dacă tasta BREAK a
                                                    fost apăsată.
056C LD-START   CALL    05E7,LD-EDGE-1              Intoarcere cu fanionul de
                JR      NC,056B,LD-BREAK            transport pe zero dacă nu
                                                    este 'tranzitie' între
                                                    aproximativ 14,000 stări T.
                                                    Dar dacă se găseste o
                                                    'tranzitie', marginea
                                                    ecranului va fi CYAN (azuriu)
```

Următorul stadiu implică o asteptare si apoi demonstrarea că semnalul pulsează.

```
                LD      HL,+0415
0574 LD-WAIT    DJNZ    0574,LD-WAIT                Lungimea acestei perioade de
                DEC     HL                          asteptare va fi de circa o
                LD      A,H                         secundă.
                OR      L
                JR      NZ,0574,LD-WAIT
                CALL    05E3,LD-EDGE-2              Se continuă numai dacă sînt
                JR      NC,056B,LD-BREAK            găsite două 'tranzitii' în
                                                    perioada de timp acordată.
```

Acum se acceptă un semnal 'conducător'.

```
0580 LD-LEADER  LD      B,+9C                       Constanta de sincronizare.
                CALL    05E3,LD-EDGE-2              Continuă numai dacă s-au
                JR      NC,056B,LD-BREAK            găsit două 'tranzitii' în
                                                    perioada de timp acordată.
                LD      A,+C6                       Totusi tranzitiile trebuie să
                CP      B                           fie găsite în aproximativ
                JR      NC,056C,LD-START            3,000 stări T pentru fiecare.
                INC     H                           Numără perechile de tranzitii
                JR      NZ,0580,LD-LEADER           în registrul H pînă găseste
                                                    '256' perechi.
```

Apoi, tactul conducător este 'off' si 'on' este parte a tactului de sincronizare.

```
058F LD-SYNC    LD      B,+C9                       Constanta de sincronizare.
                CALL    05E7,LD-EDGE-1              Fiecare tranzitie este
                JR      NC,056B,LD-BREAK            considerată pînă cînd sînt
                LD      A,B                         găsite două tranzitii
                CP      +D4                         apropiate împreună - acesta
                JR      NC,058F,LD-SYNC             va fi începutul si sfîrsitul
                                                    tranzitiei impulsului de
                                                    sincronizare 'off'.
                CALL    05E7,LD-EDGE-1              Tranzitia de sfîrsit a
                RET     NC                          impulsului 'on' trebuie să
                                                    existe.
                                                    (Revenire dacă fanionul de
                                                    transport este resetat.)
```

Octetul header-ului sau programul/setul de informatii poatet fi acum fi acum încărcat (LOAD) sau verificat (VERIFY). Dar primul octet este tipul fanionului.

```
                LD      A,C
                XOR     +03                         Culoarea marginii ecranului
                                                    de acum înainte va fi BLUE &
                                                    YELLOW (albastru & galben).
                LD      C,A
                LD      H,+00                       Initializare octet 'egalitate
                                                    paritate' pe zero.
                LD      B,+B0                       Setare constantă de
                                                    initializare pentru octetul
                                                    indicator.
                JR      05C8,LD-MARKER              Salt înainte în bucla
                                                    LOADING (încărcare) octet.
```

Bucla LOADING (încărcare) octet este folosită pentru aducerea unui octet la un moment dat. Primul octet este al fanioanelor. Acesta este urmat de octetii de informatii, iar ultimul este octetul 'paritate'.

```
05A9 LD-LOOP    EX    AF,A'F'          Se aduc fanioanele.
                JR    NZ,05B3,LD-FLAG  Salt înainte numai dacă se
                                       tratează primul octet.
                JR    NC,05BD,LD-VERIFY Salt înainte dacă se verifică
                                       o bandă.
                LD    (IX+00),L        Actuala încărcare se face la
                                       cerere.
                JR    05C2,LD-NEXT     Se face salt înainte pentru
                                       încărcarea octetului următor.
05B3 LD-FLAG    RL    C                Se păstrează temporar
                                       fanionul de transport într-un
                                       loc sigur.
                XOR   L                Revenire dacă tipul
                RET   NZ               fanionului nu coincide cu
                                       primul octet al benzii.
                                       (Fanionul de transport
                                       resetat.)
                LD    A,C              Se readuce fanionul de
                                       transport acum.
                RRA
                LD    C,A
                INC   DE               Se incrementează contorul
                JR    05C4,LD-DEC      pentru a compensa
                                       decrementarea anterioară
                                       saltului.
```

Dacă un set de informatii a fost verificat, atunci octetul actual încărcat este testat în raport cu octetul initial.

```
05BD LD-VERIFY  LD    A,(IX+00)        Se aduce octetul initial.
                XOR   L                Se compară octetul nou cu
                                       octetul initial.
                RET   NZ               Revenire dacă este
                                       'neegalitate'. (Fanionul de
                                       transport resetat.)
```

Un nou octet poate fi acum colectat de pe bandă.

```
05C2 LD-NEXT    INC   IX               Incrementare 'destinatie'.
05C4 LD-DEC     DEC   DE               Se decrementează 'contorul'.
                EX    AF,A'F'          Se salvează fanioanele.
                LD    B,+B2            Incărcare constantă de
                                       sincronizare.
05C8 LD-MARKER  LD    L,+01            Se sterge registrul 'obiect'
                                       separat printr-un bit
                                       'marcator'.
```

Bucla 'LD-8-BITS' este folosită pentru a construi în registrul L un octet.

```
05CA LD-BITS    CALL  05E3,LD-EDGE-2   Găseste lungimea impulsului
                                       'off' si 'on' al următorului
                                       bit.
                RET   NC               Revenire dacă perioada de
                                       timp a trecut. (Fanionul de
                                       transport este resetat.)
                LD    A,+CB            Se compară lungimea cu
                CP    B                aproximativ 2,400 stări T;
                                       fanionul de transport se
                                       setează pentru un '0' si se
                                       resetează pentru un '1'.
                RL    L                Se include noul bit în
                                       registrul L.
                LD    B,+B0            Se pune constanta de
                                       sincronizare pentru bitul
                                       următor.
                JP    NC,05CA,LD-8-BITS Salt înapoi dacă mai sînt
                                       biti de adus.
```

Octetul 'egalitate paritate' trebuie să fie actualizat pentru fiecare octet.

```
                LD    A,H              Se încarcă octetul 'egalitate
                XOR   L                paritate' si se include noul
                                       octet.
                LD    H,A              Se salvează încă odată.
```

Trecerile prin buclă se fac pînă cînd 'contorul' ajunge pe zero. În acest moment octetul 'egalitate paritate' poate fi tinut pe zero.

```
        LD    A,B          Dacă registrul pereche DE nu
        OR    C            este pe zero, se trece mai
        JR    NZ,05A9,LD-LOOP departe.
        LD    A,H          Se aduce octetul 'egalitate
                           paritate'.
        CP    +01          Revenire cu fanionul de
        RET                transport setat dacă valoarea
                           este zero.
                           (Fanionul de transport este
                           resetat dacă este eroare.)
```

THE 'LD-EDGE-2' AND 'LD-EDGE-1' SUBROUTINES

Aceste două subrutine formează cea mai importantă parte a operatiei de LOAD/VERIFY (încărcare/verificare).

Subrutinele sînt introduse cu o constantă de sincronizare în registrul B, iar culoarea anterioară a marginii si 'tipul tranzitiei' în registrul C.

Subrutinele revin cu fanionul de transport setat, dacă numărul de 'tranzitii' cerut a fost găsit în timpul permis; si schimbarea valorii din registrul B arată doar ce perioadă a fost necesară pentru găsirea 'tranzitiei'.

Fanionul de transport va fi resetat dacă este o eroare. Fanionul zero va fi resetat dacă semnalul 'BREAK' este tastat, sau va fi setat pentru 'timp expirat'.

Punctul de intrare LD-EDGE-2 este folosit cînd este cerută lungimea unui impuls complet, iar LD-EDGE-1 este folosit pentru a găsi timpul dinaintea 'tranzitiei' următoare.

```
05E3 LD-EDGE-2   CALL  05E7,LD-EDGE-1   Prin urmare, apelează
                                        LD-EDGE-1 de două ori;
                 RET   NC               revenire dacă este eroare.
05E7 LD-EDGE-1   LD    A,+16            Asteaptă 358 stări T înaintea
05E9 LD-DELAY    DEC   A                intrării în bucla esantion.
                 JR    NZ,05E9,LD-DELAY
                 AND   A
```

Intrarea în bucla esantion s-a făcut. Valoarea din registrul B se incrementează pentru fiecare trecere; 'timpul expirat' este dat cînd B ajunge pe zero.

```
05EB LD-SAMPLE   INC   B              Numără fiecare trecere.
                 RET   Z              Revenire cu fanionul de
                                      transport resetat si cu
                                      fanionul zero setat dacă este
                                      'timpul expirat'.
                 LD    A,+7F          Se citeste din portul +7FFE.
                 IN    A,(+FE)        Acesta este BREAK & EAR.
                 RRA                  Roteste octetul.
                 RET   NC             Revenire cu fanionul de
                                      transport resetat si cu
                                      fanionul zero resetat dacă
                                      s-a tastat 'BREAK'.
                 XOR   C              Acum se testează octetul față
                 AND   +20            de 'tipul ultimei tranzitii';
                 JR    Z,05EB,LD-SAMPLE salt înapoi numai dacă s-a
                                      schimbat.
```

O nouă 'tranzitie' a fost găsită în perioada de timp alocată căutării. Deci se schimbă culoarea marginii si se setează fanionul de transport.

```
                 LD    A,C            Se schimbă 'tipul ultimei
                 CPL                  tranzitii' si culoarea
                                      marginii ecranului.
                 LD    C,A
                 AND   +07            Se păstrează numai culoarea
                                      marginii.
                 OR    +08            Semnal 'MIC off'.
                 OUT   (+FE),A        Se schimbă culoarea marginii
                                      (RED/CYAN (rosu/azuriu) sau
                                      BLUE/ YELLOW (albastru /
                                      galben)).
                 SCF                  Se semnalizează reusita
                 RET                  căutării înaintea revenirii.
```

Notă: Subrutina LD-EDGE-1 ia 465 stări T, plu 58 stări T pentru fiecare trecere nereusită prin bucla esantion.

De exemplu, de aceea, cînd se asteaptă impulsusul sincron (vezi LD-SYNC la 058F) se incrementează zece treceri aditionale prin bucla esantion. Căutarea este astfel pentru ca următoarea tranzitie să fie găsită, grosolon, în 1,100 stări T (465 + 10 * 58 + deasupra). Aceasta va dovedi realizarea impulsului sincron 'off' care vine după lungul 'impuls conducător'.

THE 'SAVE, LOAD, VERIFY & MERGE' COMMAND ROUTINES
(RUTINELE DE SALVARE, INCARCARE, VERIFICARE SI UNIRE)

Punctul de intrare SAVE-ETC este folosit pentru toate cele patru comenzi. Valoarea continută în T-ADDR oricum va fi distinctă pentru cele patru comenzi. Prima parte a rutinei următoare se ocupă de construirea unui 'header informatie' în spatiul de lucru.

```
0605 SAVE-ETC     POP   AF                  Se trimite adresa SCAN-LOOP.
                  LD    A,(T-ADDR-lo)       Se scade T-ADDR-lo cu +E0;
                  SUB   +E0                 se dă +00 pentru SAVE, +01
                  LD    (T-ADDR-lo),A       pentru LOAD, +02 pentru
                                            VERIFY si +03 pentru MERGE.
                  CALL  1C8C,EXPT-EXP       Se trec parametrii 'nume' în
                                            stiva calculatorului.
                  CALL  2530,SYNTAX-Z       Salt înainte dacă se caută
                  JR    Z,0652,SA-DATA      sintaxa.
                  LD    BC,+0011            Se cedează 17 locatii pentru
                  LD    A,(T-ADDR-lo)       header-ul SAVE, dar 34 pentru
                  AND   A                   celelalte comenzi.
                  JR    Z,0621,SA-SPACE
                  LD    C,+22
0621 SA-SPACE     RST   0030,BC-SPACES      Spatiul total este cerut din
                                            spatiul de lucru.
                  PUSH  DE                  Se copiază adresa de start
                  POP   IX                  din registrul pereche IX.
                  LD    B,+0B               Numele unui program poate
                  LD    A,+20               avea pînă la zece caractere,
0629 SA-BLANK     LD    (DE),A              dar prima dată se introduc 11
                  INC   DE                  caractere spatiu în zona
                  DJNZ  0629,SA-BLANK       rezervată.
                  LD    (IX+01),+FF         Un nume invalidat este numai
                                            +FF.
                  CALL  28F1,STK-FETCH      Parametrii numelui sînt
                                            încărcati si lungimea este
                                            testată.
                  LD    HL,+FFF6            Este '-10'.
                  DEC   BC                  Da, salt înainte dacă
                  ADD   HL,BC               lungimea numelui nu este prea
                  INC   BC                  lungă (nu mai mult de 10
                  JR    NC,0643,SA-NAME     caractere).
                  LD    A,(T-ADDR-lo)       Dar aprobă încărcarea (LOAD),
                  AND   A                   verificarea (VERIFY) si
                  JR    NZ,0644,SA-NULL     unirea (MERGE) programelor cu
                                            nume 'invalidat' sau foarte
                                            lung.
```

Prezentarea 1 - numele invalidat al fisierului.

```
0642 REPORT-F     RST   0008,ERROR-1        Apel rutină pentru tratarea
                  DEFB  +0E                 erorii.
```

Se continuă tratarea numelui programului.

```
0644 SA-NULL      LD    A,B                 Salt înainte dacă numele are
                  OR    C                   lungimea invalidată.
                  JR    Z,0652,SA-DATA
                  LD    BC,+000A            Dar se trunchiază numele prea
                                            lungi.
```

Numele va fi transferat acum în spatiul de lucru (a doua locatie în continuare).

```
0643 SA-NAME      PUSH  IX                  Se aduce adresa de început în
                  POP   HL                  registrul pereche HL.
                  INC   HL                  Pas spre a doua locatie.
                  EX    DE,HL               Se schimbă indicatorul si se
                  LDIR                      copiază numele.
```

Se prezintă în continuare cazul în care parametrii multi si diferiti urmează o comandă.
Se începe cu tratarea 'xxx''name''DATA'.

```
0652 SA-DATA    RST   0018,GET-CHAR      Codul prezent este simbolul
                CP    +E4               'DATA'?
                JR    NZ,06A0,SA-SCR$   Dacă nu este, salt.
                LD    A,(T-ADDR-lo)     Oricum, nu este posibil să
                CP    +03               fie 'MERGE name DATA'.
                JP    Z,1C8A,REPORT-C
                RST   0020,NEXT-CHAR    Se avansează CH-ADD.
                CALL  28B2,LOOK-VARS    Se face controlul dispunerii
                                        în spatiul variabilelor.
                SET   7,C               Setare bit 7 în dispunerea
                                        numelui.
                JR    NC,0672,SA-V-OLD  Salt dacă se tratează o
                                        dispunere existentă.
                LD    HL,+0000          Semnal 'utilizare dispunere
                                        nouă'.
                LD    A,(T-ADDR-lo)     Se încarcă valoarea T-ADDR si
                DEC   A                 este eroare dacă se încearcă
                JR    Z,0685,SA-V-NEW   sau verificare a unei noi
                                        dispuneri.
```

Prezentarea 2 - Variabilă negăsită.

```
0670 REPORT-2   RST   0008,ERROR-1      Apel rutină tratare eroare.
                DEFB  +01
```

Se continuă cu tratarea unei dispuneri existente.

```
0672 SA-V-OLD   JP    NZ,1C8A,REPORT-C  Notă: Acesta nu reuseste să
                                        excludă sirurile simple.
                CALL  2530,SYNTAX-Z     Salt înainte dacă se verifică
                JR    Z,0692,SA-DATA-1  sintaxa.
                INC   HL                Incrementare 'lungime mică' a
                                        variabilei.
                LD    A,(HL)            Octetul lungimii mici trece
                LD    (IX+0B),A         în spatiul de lucru; el este
                INC   HL                urmat de octetul lungime
                                        mare.
                LD    A,(HL)
                LD    (IX+0C),A
                INC   HL                Pas peste lungimea octetului.
```

Următoarea parte este comună si pentru dispunerea 'veche' si pentru dispunerea 'nouă'. Notă: Cale eronată sintactic.

```
0685 SA-V-NEW   LD    (IX+0E),C         Incărcare nume dispunere.
                LD    A,+01             Initializare dispunere de
                                        numere.
                BIT   6,C               Salt dacă este asa.
                JR    Z,068F,SA-V-TYPE
                INC   A                 Este o dispunere de
                                        caractere.
068F SA-V-TYPE  LD    (IX+00),A         Salvare 'tip' în prima
                                        locatie a spatiului de
                                        header.
```

Ultima parte a afirmatiei este examinată înaintea celorlalte căi.

```
0692 SA-DATA-1  EX    DE,HL             Salvare indicator în DE.
                RST   0020,NEXT-CHAR    Următorul caracter este')' ?
                CP    +29
                JR    NZ,0672,SA-V-OLD  Se prezintă registrul C dacă
                                        nu este.
                RST   0020,NEXT-CHAR    Avansează CH-ADD.
                CALL  1BEE,CHECK-END    Inscrie următoarea formulare
                                        dacă verifică sintaxa.
                EX    DE,HL             Readucere indicator în
                JP    075A,SA-ALL       registrul pereche HL înainte
                                        de saltul în fată.
                                        (Indicatorul arată începutul
                                        unei dispuneri cu continut
                                        existent.)
```

Acum se va considera 'SCREEN$'.

```
06A0 SA-SCR$    CP    +AA               Codul prezent este 'SCREEN$'.
                JR    NZ,06C3,SA-CODE   Salt dacă nu este.
                LD    A,(T-ADDR-lo)     Oricum nu este posibil să fie
                CP    +03               'MERGE name SCREEN$'.
```

```
                JP    Z,1C8A,REPORT-C
                RST   0020,NEXT-CHAR          Avanseazà CH-ADD.
                CALL  1BEE,CHECK-END          Se înscrie următoarea
                                              formulare dacà verificà
                                              sintaxa.
                LD    (IX+0B),+00             Spatiul pentru înscriere si
                LD    (IX+0C),+1B             spatiul atributelor ocupà
                                              +1B00 locatii, care încep la
                LD    HL,+4000                +4000; aceste detalii sînt
                LD    (IX+0D),L               trecute în spatiul headerului
                LD    (IX+0E),H               din spatiul de lucru.
                JR    0710,SA-TYPE-3          Salt înainte.

Acum se va considera 'CODE'.

06C3 SA-CODE    CP    +AF                     Este codul prezentat simbolul
                                              'CODE' ?
                JR    NZ,0716,SA-LINE         Dacà nu este, atunci salt.
                LD    A,(T-ADDR-lo)           Oricum nu este posibil sà fie
                CP    +03                     'MERGE name CODE'.
                JP    Z,1C8A,REPORT-C
                RST   0020,NEXT-CHAR          Avanseazà CH-ADD.
                CALL  2048,PR-ST-END          Salt înainte dacà formularea
                JR    NZ,06E1,SA-CODE-1       nu a fost terminatà.
                LD    A,(T-ADDR-lo)           Oricum nu este posibil sà fie
                AND   A                       'SAVE name CODE' cu el
                JP    A,1C8A,REPORT-C         însusi.
                CALL  1CE6,USE-ZERO           Se pune un zero în stiva
                                              calculatorului pentru 'start'
                JR    06F0,SA-CODE-2          Salt înainte.

Prezentare 'adresà de început.

06E1 SA-CODE-1  CALL  1C82,EXPT-1NUM          Se aduce primul numàr.
                RST   0018,GET-CHAR           Este caracterul adus ',' ?
                CP    +2C
                JR    Z,06F5,SA-CODE-3        Dacà este, salt înainte -
                                              numàrul a fost 'adresa de
                                              început'.
                LD    A,(T-ADDR-lo)           Oricum se refuzà 'SAVE name
                AND   A                       CODE', care nu are un
                JP    Z,1C8A,REPORT-C         'început' si o 'lungime'.
06F0 SA-CODE-2  CALL  1CE6,USE-ZERO           Se pune un zero în stiva
                                              calculatorului pentru
                                              'lungime'.
                JR    06F9,SA-CODE-4          Salt înainte.

Se încarcà 'lungimea' asa cum a fost specificat.

06F5 SA-CODE-3  RST   0020,NEXT-CHAR          Avanaseazà CH-ADD.
                CALL  1C82,EXPT-1NUM          Se încarcà 'lungimea'.

Parametrii sînt acum pàstrati în spatiul header-ului din spatiul de lucru.

06F9 SA-CODE-4  CALL  1BEE,CHECK-END          Deplasare în formularea
                                              următoare care verificà acum
                                              sintaxa.
                CALL  1E99,FIND-INT2          Se condenseazà 'lungimea' în
                LD    (IX+0B),C               registrul pereche BC si se
                LD    (IX+0C),B               stochează.
                CALL  1E99,FIND-INT2          Se condenseazà 'adresa de
                LD    (IX+0D),C               început' în registrul pereche
                LD    (IX+0E),B               BC si se stochează.
                LD    H,B                     Se transferà 'indicatorul' în
                LD    L,C                     registrul pereche HL, ca de
                                              obicei.

'SCREEN$' si 'CODE' sînt ambele de tipul 3.

0710 SA-TYPE-3  LD    (IX+00),+03             Se încarcà tipul numàrului.
                JR    075A,SA-ALL             Se revine la alte cài.

Acum se vor prezenta 'LINE' si 'nu mai multi parametrii'.

0716 SA-LINE    CP    +CA                     Codul prezent este 'LINE' ?
                JR    Z,0723,SA-LINE          Dacà este, atunci se executà
                                              salt.
                CALL  1BEE,CHECK-END          Dacà se verificà sintaxa,
                                              atunci salt la formularea
```

```
                                                  următoare.
                LD    (IX+0E),+80                 Dacă nu sînt mai multi
                                                  parametri se introduce +80.
                JR    073A,SA-TYPE-0              Salt înainte.
```

Incărcare 'număr linie', ce trebuie să urmeze 'LINE'.

```
0723 SA-LINE-1  LD    A,(T-ADDR-lo                Este permis doar 'SAVE name
                AND   A                           LINE number'.
                JP    NZ,1C8A,REPORT-C
                RST   0020,NEXT-CHAR              Avansează CH-ADD.
                CALL  1C82,EXPT-1NUM              Se trece numărul în stiva
                                                  calculatorului.
                CALL  1BEE,CHECK-END              Dacă se verifică sintaxa,
                                                  salt în formularea următoare.
                CALL  1E99,FIND-INT2              Se comprimă 'număr linie' în
                LD    (IX+0D),C                   registrul pereche BC si se
                LD    (IX+0E),B                   stochează.
```

'LINE si 'nu mai multi parametri' sînt de tip 0.

```
073A SA-TYPE-0  LD    (IX+00),+00                 Se încarcă 'tipul' numărului.
```

Parametrii care descriu programul si variabilele lui sînt păstrate si stocate în spatiul header-ului din spatiul de lucru.

```
                LD    HL,(E-LINE)                 Indicatorul pentru sfîrsitul
                                                  spatiului variabilelor.
                LD    DE,(PROG)                   Indicatorul pentru începutul
                                                  programului BASIC.
                SCF                               Acum se execută scăderi
                SBC   HL,DE                       pentru a găsi lungimea
                LD    (IX+0B),L                   'program & variabile';
                LD    (IX+0C),H                   rezultatul se stochează.
                LD    HL,(VARS)                   Operatia se repetă, dar de
                SBC   HL,DE                       data aceasta se stochează
                LD    (IX+0E),L                   doar lungimea 'program'.
                LD    (IX+10),H
                EX    DE,HL                       Se transferă 'indicatorul' în
                                                  registrul pereche HL, ca de
                                                  obicei.
```

In toate cazurile header-ul de informatie a fost pregătit.

Locatia 'IX+00' contine numărul tipului.
Locatiile 'IX+01 până la IX+0A' contin numele (+FF în 'IX+01' dacă este invalidat.)
Locatiile 'IX+0B & IX+0C' contin numărul de octeti care trebuie să fie găsiti în 'blocul de informatii'.
Locatiile 'IX+0D până la IX+10' contin o varietate de parametrii a căror interpretare depinde de 'tip'.

Rutina continuă cu primul task realizat astfel încît să separe SAVE de LOAD, VERIFY si MERGE.

```
075A SA-ALL     LD    A,(T-ADDR-lo)               Salt înainte dacă se tratează
                AND   A                           o comandă SAVE (salvare).
                JP    Z,0970,SA-CONTRL
```

In cazul unei comenzi LOAD, VERIFY sau MERGE, primii 17 octeti ai spatiului header-ului din spatiul de lucru contin informatia pregătită, descrisă mai sus; acum este momentul să se aducă 'header-ul' de pe bandă.

```
                PUSH  HL                          Salvare indicator
                                                  'destinatie'.
                LD    BC,+0011                    Din registrul dublu IX se
                ADD   IX,BC                       aduce adresa de bază pentru
                                                  'al doilea spatiu al
                                                  header-ului'.
```

Acum intervine o buclă; se păstrează bucla numai cînd 'header-ul' a fost încărcat.

```
0767 LD-LOOK-H  PUSH  IX                          Se face o copie a adresei de
                                                  bază.
                LD    DE,+0011                    Se încarcă 17 octeti.
                XOR   A                           Semnal 'header'.
                SCF                               Semnal 'LOAD'.
                CALL  0556,LD-BYTES               Acum caută header.
```

```
                POP   IX                    Se regăseste adresa de bază.
                JR    NC,0767,LD-LOOK-H     Ciclează bucla pînă se cînd
                                            se termină încărcarea.
```

Noul 'header' este afisat pe ecran, dar rutina se va executa numai dacă 'noul' header coincide cu 'vechiul' header.

```
                LD    A,+FE                 Se asigură că este deschis
                CALL  1601,CHAN-OPEN        canalul 'S'.
                LD    (SCR-CT),+03          Setare numărător defilare.
                LD    C,+80                 Semnal 'numele nu coincid'.
                LD    A,(IX+00)             Se compară 'noul' tip cu
                CP    (IX-11)               'vechiul' tip.
                JR    NZ,078A,LD-TYPE       Salt dacă tipurile nu coincid
                LD    C,+F6                 Dacă ele coincid, va apare
                                            semnal 'zece caractere
                                            coincid'.
078A LD-TYPE    CP    +04                   Curătarea 'header-ului' nu
                JR    NC,0767,LD-LOOK-H     are sens dacă este 'tip 4 sau
                                            mai mult'.
```

Va fi tipărit mesajul potrivit - 'Program', 'Număr spatiu', 'Spatiu caracteri' sau 'Octeti'.

```
                                            Adresa de bază a blocului de
                                            mesaj.
                                            Se salvează registrul C pînă
                                            cînd mesajul potrivit va fi
                                            tipărit.
```

'Nume nou' este tipărit si asa cum sînt date, 'nou' si 'vechi' nume sînt comparate.

```
                PUSH  IX
                LD    DE,+09C0
                POP   DE
                LD    HL,+FFF0
                PUSH  BC
                ADD   HL,DE
                CALL  0C0A,PO-MSG
                LD    B,+0A
                POP   BC
                LD    A,(HL)                Salt înainte dacă nu este
                INC   A                     coincidentă cu noul nume.
                JR    NZ,07A6,LD-NAME
                LD    A,C                   Dacă 'nume vechi' este
                ADD   A,B                   invalidat, este trimis
                LD    C,A                   semnalul 'zece caractere
                                            deja coincid'.
```

Este introdusă o buclă pentru tipărirea caracterelor de la 'nume nou'. Numele va fi acceptat dacă 'contorul' ajunge pe zero, pînă la urmă.

```
07A6 LD-NAME    INC   DE                   Se consideră că fiecare
                LD    A,(DE)                caracter de la 'nume nou'
                                            este unul după altul.
                CP    (HL)                  Se compară cu caracterul
                INC   HL                    corespunzător al lui 'nume
                                            vechi'.
                JR    NZ,07AD,LD-CH-PR      Nu se numără dacă nu coincid.
                INC   C
07AD LD-CH-PR   RST   0010,PRINT-A-1        Se tipăreste 'noul' caracter.
                DJNZ  07A6,LD-NAME          Se buclează pentru zece
                                            caractere.
                BIT   7,C                   Se acceptă numele numai dacă
                JR    NZ,0767,LD-LOOK-H     contorul a ajuns la zero.
                LD    A,+0D                 Urmează 'nume nou' cu
                RST   0010,PRINT-A-1        revenire la începutul
                                            rîndului (CR).
```

Headerul corect a fost găsit si a sosit momentul pentru a fi prezentate separat cele trei comenzi LOAD (încărcare), VERIFY (verificare) si MERGE (unire).

```
                POP   HL                    Incarcă indicatorul.
                LD    A,(IX+00)             'SCREEN$ si CODE' sînt
                CP    +03                   comparate cu VERIFY.
                JR    Z,07CB,VR-CONTRL
                LD    A,(T-ADDR-lo)         Salt înainte dacă este
```

```
                DEC   A                    folosită o comandă LOAD.
                JP    Z,0808,LD-CONTRL
                CP    +02                  Salt înainte dacă este
                JP    Z,08B6,ME-CONTRL     folosită o comandă MERGE;
                                           se continuă cu o comandă
                                           VERIFY.
```

THE 'VERIFY' ROUTINE (RUTINA DE VERIFICARE)
Procesul de verificare implică încărcarea (LOAD) unui bloc de informatii, un octet la un moment dat, dar octetii nu sînt retinuti - ci doar verificati. Această rutină este folosită de asemenea pentru încărcarea (LOAD) blocurilor de informatii care au fost descrise cu 'SCREEN$ & CODE'.

```
07CB VR-CONTRL  PUSH  HL                   Salvare 'indicator'.
                LD    L,(IX-06)            Se aduce 'număr octeti' cum
                LD    H,(IX-05)            s-a descris în 'vechiul'
                                           header.
                LD    E,(IX+0B)            Se aduce si numărul din
                LD    D,(IX+0C)            'noul' header
                LD    A,H                  Salt înainte dacă 'lungimea'
                OR    L                    nu este specificată.
                JR    Z,07E9,VR-CONT-1     De exemplu, doar 'LOAD nume
                                           CODE'.
                SBC   HL,DE                Se dă prezentarea R dacă se
                JR    C,0806,REPORT-R      încearcă încărcarea (LOAD)
                                           unui bloc mai mare decît a
                                           fost cerut.
                JR    Z,07E9,VR-CONT-1     Acceptă 'lungimi' egale.
                LD    A,(IX+00)            De asemenea, se dă
                CP    +03                  prezentarea R dacă se
                JR    NZ,0806,REPORT-R     încearcă verificarea
                                           (VERIFY) blocurilor de
                                           mărimi diferite. ('Lungime
                                           veche' este mai mare decît
                                           'lungime nouă'.)
```

Rutina continuă, luînd în considerare 'indicatorul destinatie'.

```
07E9 VR-CONT-1  POP   HL                   Se încarcă 'indicatorul';
                                           acesta este 'start'.
                LD    A,H                  Acest 'indicator va fi
                OR    L                    folosit numai în cazul în
                JR    NZ,07F4,VR-CONT-2    care zero, caz în care
                LD    L,(IX+0D)            'start' găsit în 'noul'
                LD    H,(IX+0E)            header va fi folosit pe loc.
```

Indicatorul pentru VERIFY/LOAD (verificare/încărcare) este luat acum în considerare si încărcarea actuală făcută.

```
07F4 VR-CONT-2  PUSH  HL                   Se mută 'indicatorul' în
                POP   IX                   registrul pereche HL.
                LD    A,(T-ADDR-lo)        Salt înainte numai dacă se
                CP    +02                  foloseste comanda VERIFY;
                SCF                        fanionul de transport va
                JR    NZ,0800,VR-CONT-3    indica 'LOAD' (încărcare).
                AND   A                    Semnalare 'VERIFY'.
0800 VR-CONT-3  LD    A,+FF                Se semnalizează 'acceptare
                                           numai bloc de informatii'
                                           înaintea încărcării (LOAD)
                                           blocului.
```

THE 'LOAD DATA BLOCK' SUBROUTINE (SUBRUTINA 'INCARCARE BLOC DE INFORMATII')
Subrutina este comună pentru toate rutinele de încărcare (LOAD). În cazul LOAD & VERIFY (încărcare si verificare) subrutina actionează ca o revenire completă de pe caseta de tratare a rutinei, dar în cazul MERGE (unire) blocul de informatii trebuie să fie 'contopit'.

```
0802 LD-BLOCK   CALL  0556,LD-BYTES        Încărcarea si verificarea
                                           unui bloc de informatii.
                RET   C                    Revenire numai în caz de
                                           eroare.
```

Prezentare R - Inregistrează încărcare eronată.

```
0806 REPORT-R   RST   0008,ERROR-1         Apel rutină tratare eroare.
                DEFB  +1A
```

THE 'LOAD' CONTROL ROUTINE (RUTINA DE CONTROL INCARCARE)
Această rutină controlează încărcarea unui program BASIC, si a variabilelor

lui, sau a unui spatiu.

```
0808 LD-CONTRL   LD   E,(IX+0B)        Incărcare număr octeti' asa
                 LD   D,(IX+0C)        cum sînt dati în 'noul
                                       header'.
                 PUSH HL               Se salveaza 'indicator
                                       destinatie'.
                 LD   A,H              Salt înainte numai dacă se
                 OR   L                încearcă încărcarea unei
                 JR   NZ,0819,LD-CONT-1 portiuni anterioare
                                       nedeclarate.
                 INC  DE               Se adună trei biti la
                 INC  DE               lungime - pentru nume,
                 INC  DE               lungimea minimă si lungimea
                 EX   DE,HL            maximă a unei variabile.
                 JR   0825,LD-CONT-2   Salt înainte.
```

Acum se evaluează dacă sînt suficiente locatii de memorie pentru noul bloc de informatii.

```
0819 LD-CONT-1   LD   L,(IX-06)        Se încarcă mărimea 'program
                 LD   H,(IX-05)        + variabile sau spatiu'
                                       existente.
                 EX   DE,HL
                 SCF                   Salt înainte dacă nu se cer
                 SBC  HL,DE            prea multe locatii, tinînd
                 JR   C,082E,LD-DATA   cont de gradul prezent de
                                       folosire al memoriei.
```

Se execută actualul test pentru locatii.

```
0825 LD-CONT-2   LD   BC,+0005         Se acceptă deasupra un spatiu
                 ADD  HL,DE            de cinci octeti.
                 LD   B,R              Se mută rezultatul în
                 LD   C,L              registrul pereche BC si se
                 CALL 1F05,TEST-ROOM   execută testul.
```

Acum se prezintă partea cu încărcarea unei portiuni.

```
082E LD-DATA     POP  HL               Se readuce 'indicatorul'.
                 LD   A,(IX+00)        Salt înainte dacă se încarcă
                 AND  A                un program BASIC.
                 JR   Z,0873,LD-PROG
                 LD   A,H              Salt înainte dacă se încarcă
                 OR   L                o portiune nouă
                 JR   Z,084C,LD-DATA-1
                 DEC  HL               Se încarcă 'lungimea' unei
                 LD   B,(HL)           portiuni existente prin
                 DEC  HL               adunarea octetilor din
                 LD   C,(HL)           spatiul variabilelor.
                 DEC  HL               Se indică vechiul nume.
                 INC  BC               Se adună trei biti la lungime
                 INC  BC               - unul pentru nume si doi
                 INC  BC               pentru 'lungime'.
                 LD   (X-PRT),IX       Se salvează registrul pereche
                 CALL 19E8,RECLAIM-2   IX până când se cere 'vechea'
                 LD   IX,(X-PTR)       portiune.
```

Spatiul este acum accesibil pentru noua portiune - la sfîrsitul spatiului variabilelor prezente.

```
084C LD-DATA-1   LD   HL,(E-LINE)      Găseste indicatorul la
                 DEC  HL               sfîrsitul ariei variabilelor
                                       - al'80-lea bit'.
                 LD   C,(IX+0B)        Se încarcă 'lungimea' noii
                 LD   B,(IX+0C)        portiuni.
                 PUSH BC               Salvarea acestei 'lungimi'.
                 INC  BC               Se adaugă trei biti -unul
                 INC  BC               pentru nume si doi pentru
                 INC  BC               'lungime'.
                 LD   A,(IX-03)        'IX+0E' din vechiul header
                                       dă numele portiunii.
                 PUSH AF               Numele este salvat până când
                 CALL 1655,MAKE-ROOM   numărul potrivit de locatii
                 INC  HL               este făcut accesibil. De fapt
                 POP  AF               'BC' spatii înaintea 'noilor
                                       80 octeti'.
                 LD   (HL),A           Numele este introdus.
                 POP  DE               'Lungimea' este adusă si cei
```

```
                INC   HL                    doi octeti sint de asemenea
                LD    (HL),E                introdusi.
                INC   HL
                LD    (HL),D
                INC   HL                    HL indică acum prima locatie
                                            care urmează a fi înlocuită
                                            cu informatii de pe bandă.
                PUSH  HL                    Această adresă este mutată în
                POP   IX                    registrul pereche IX;
                SCF                         fanionul de transport este
                LD    A,+FF                 setat; este semnalat 'bloc de
                JP    0802,LD-BLOCK         informatii'; si blocul este
                                            încărcat.
```

Acum se prezintă partea cu încărcarea unui program BASIC cu variabilele lui.

```
0873 LD-PROG    EX    DE,HL                 Se salvează 'indicatorul
                                            destinatie'.
                LD    HL,(E-LINE)           Găseste adresa indicatorului
                DEC   HL                    de sfîrsit a portiunii
                                            curente de variabile - cei
                                            '80-octeti'.
                LD    (X-PTR),IX            Salvare temporară a
                                            registrului IX.
                LD    C,(IX+0B)             Se aduce 'lungimea' noului
                LD    B,(IX+0C)             bloc de informatii.
                PUSH  BC                    Se păstrează o copie a
                CALL  19E5,RECLAIM-1        'lungimii' pînă cînd
                POP   BC                    programul prezent si aria
                                            variabilelor sint cerute.
                PUSH  HL                    Se salvează indicatorul în
                PUSH  BC                    spatiul programului si
                                            lungimea noului bloc de
                                            informatii.
                CALL  1655,MAKE-ROOM        Se accesează un număr
                                            suficient de locatii pentru
                                            noul program si variabilele
                                            lui.
                LD    IX,(X-PTR)            Se readuce continutul
                                            registrului pereche IX.
                INC   HL                    De asemenea trebuie pregătit
                LD    C,(IX+0F)             sistemul de variabile VARS
                LD    B,(IX+10)             pentru noul program.
                ADD   HL,BC
                LD    (VARS),HL
                LD    H,(IX+0E)             Dacă un număr de linie a fost
                LD    A,H                   specificat, atunci si acesta
                AND   +C0                   trebuie luat în considerare.
                JR    NZ,08AD,LD-PROG-1     Salt dacă 'nici un număr';
                LD    L,(IX+0D)             altfel se setează NEWPPC &
                LD    (NEWPPC),HL           NSPPC.
                LD    (NSPPC),+00
```

Acum blocul de informatii poate fi încărcat.

```
08AD LD-PROG-1  POP   DE                    Se aduce 'lungimea'.
                POP   IX                    Se aduce 'începutul'.
                SCF                         Semnal 'LOAD' (încărcare).
                LD    A,+FF                 Semnal 'numai bloc de
                                            informatii'.
                JP    0802,LD-BLOCK         Acum se încarcă (LOAD).
```

THE 'MERGE' CONTROL ROUTINE (RUTINA DE CONTROL UNIRE)
Sînt trei părti principale în această rutină.

i. LOAD (încărcare) bloc de informatii în spatiul de lucru.
ii. MERGE (contopirea) liniilor noului program cu vechiul program.
iii. MERGE (contopirea) noilor variabile cu cele vechi.

De aceea se începe cu încărcarea blocului de informatii.

```
08B6 ME-CONTROL LD    C,(IX+0B)             Se aduce 'lungimea' blocului
                LD    B,(IX+0C)             de informatii.
                PUSH  BC                    Se salvează o copie a
                                            'lungimii'.
                INC   BC                    Acum se fac 'lungime + 1'
                RST   0030,BC-SPACES        locatii accesibile în spatiul
                                            de lucru.
                LD    (HL),+80              Se pune un indicator de
```

```
                    EX    DE,HL          sfîrsit la extrema locatiei.
                                         Se mută indicatorul 'start'
                                         în registrul pereche HL.
                    POP   DE             Se aduce 'lungimea' inițială.
                    PUSH  HL             Se salvează o copie a lui
                                         'start'.
                    PUSH  HL             Se setează registrul pereche
                    POP   IX             IX pentru încărcarea actuală.
                    SCF                  Semnal 'LOAD' (încărcare).
                    LD    A,+FF          Semnal 'numai bloc de
                                         informatii'.
                    CALL  0802,LD-BLOCK  Se încarcă blocul de
                                         informatii.
```

Liniile noului program sînt unite (MERGE) cu liniile vechiului program.

```
                    POP   HL             Se aduce 'începutul' noului
                                         program.
                    LD    DE,(PROG)      Se initializează DE cu
                                         'începutul' ('start')
                                         vechiului program.
```

Se intră într-o buclă pentru partea cu liniile noului program.

```
08D2 ME-NEW-LP      LD    A,(HL)         Se aduce un număr de linie si
                    AND   +C0            se testează.
                    JR    NZ,08F0,ME-VAR-LP  Salt cînd s-au terminat toate
                                         liniile.
```

Intrare în interiorul buclei pentru partea cu liniile vechiului program.

```
08D7 ME-OLD-LP      LD    A,(DE)         Se încarcă octetul ce contine
                    INC   DE             numărul maxim de linii si se
                                         compară.
                    CP    (HL)           Salt înainte dacă nu este
                    INC   HL             coincidentă, dar în orice caz
                    JR    NZ,08DF,ME-OLD-L1  vor avansa ambele indicatoare
                    LD    A,(DE)         Se repetă comparatia pentru
                    CP    (HL)           octetul care contine numărul
                                         minim de linii.
08DF ME-OLD-L1      DEC   DE             Acum se decrementează
                    DEC   HL             indicatorul.
                    JR    NC,08EB,ME-NEW-L2  Salt înainte dacă s-a găsit
                                         locul corect pentru linia
                                         noului program.
                    PUSH  HL             Altfel trebuie găsită adresa
                    EX    DE,HL          începutului unei alte linii
                    CALL  19B8,NEXT-ONE  vechi.
                    POP   HL
                    JR    08D7,ME-OLD-LP Se ciclează bucla pentru
                                         fiecare 'linie veche'.
08EB ME-NEW-L2      CALL  092C,ME-ENTER   Se introduce 'linia nouă' si
                    JR    08D2,ME-NEW-LP se ciclează din nou bucla
                                         exterioară.
```

In mod analog, variabilele noului program sînt unite (MERGE) cu variabilele vechiului program.
O buclă este introdusă pentru rezolvarea fiecărei noi variabile, una cîte una.

```
08F0 ME-VAR-LP      LD    A,(HL)         Se aduce fiecare nume de
                    LD    C,A            variabilă unul cîte unul si
                                         se testează.
                    CP    +80            Revenire dacă au fost testate
                    RET   Z              toate variabilele.
                    PUSH  HL             Se salvează valoarea actuală
                                         a indicatorului curent.
                    LD    HL,(VARS)      Se aduce VARS (pentru vechiul
                                         program).
```

Acum se intră într-o buclă internă pentru căutarea spatiului variabilelor existente.

```
08F9 ME-OLD-VP      LD    A,(HL)         Se aduce fiecare nume de
                    CP    +80            variabilă si se testează.
                    JR    Z,0923,ME-VAR-L2  Salt înainte îndată ce s-a
                                         găsit indicatorul de sfîrsit.
                                         (Face o adăugare).
                    CP    C              Se compară numele (primul
                                         octet).
```

```
                JR    Z,0909,ME-OLD-V2      Salt înainte pentru a-l lua
                                            în considerare mai tîrziu;
                                            aici se va face revenire
                                            dacă se dovedeste că numele
                                            nu coincid în totalitate.
0901 ME-OLD-V1  PUSH  BC                    Se salvează noul nume al
                CALL  1988,NEXT-ONE         variabilei pînă cînd
                POP   BC                    următoarea 'variabilă veche'
                                            este locatată.
                EX    DE,HL                 Se readuce indicatorul în
                JR    08F9,ME-OLD-VP        registrul pereche DE si se
                                            ciclează din nou bucla.
```

Variabilele vechi si noi coincid prin respectarea primului octet, dar variabilele cu nume lung necesită o totală coincidentă.

```
0909 ME-OLD-V2  AND   +E0                   Se consideră doar bitii 7, 6
                                            si 5.
                CP    +A0                   Se acceptă toate tipurile de
                JR    NZ,0921,ME-VAR-L1     variabile, exceptînd
                                            'variabilele cu nume lung'.
                POP   DE                    Se face DE purtătorul
                PUSH  DE                    primului caracter al 'noului
                                            nume'.
                PUSH  HL                    Se salvează indicatorul
                                            'vechiului nume'.
```

Se intră într-o buclă de comparare a literelor numelor lungi.

```
0912 ME-OLD-V3  INC   HL                    Se actualizează indicatorii
                INC   DE                    'vechi' si 'nou'.
                LD    A,(DE)                Se compară cele două litere.
                CP    (HL)
                JR    NZ,091E,ME-OLD-V4     Salt înainte dacă literele nu
                                            coincid.
                RLA                         Se ciclează bucla pînă cînd
                JR    NC,0912,ME-OLD-V3     se găseste 'ultimul caracter'
                POP   HL                    Se aduce indicatorul la
                                            începutul 'vechiului' nume si
                JR    0921,ME-VAR-L1        salt înainte - reusit.
091E ME-OLD-V4  POP   HL                    Se aduce indicatorul si
                JR    0901,ME-OLD-V1        salt înapoi - nereusit.
```

Se vine aici dacă s-a găsit coincidentă.

```
0921 ME-VAR-L1  LD    A,+FF                 Se semnalează 'înlocuire'
                                            variabilă.
```

Si se vine aici dacă nu s-a găsit coincidentă. (O reținere +80 - variabilă ce trebuie 'adăugată').

```
0923 ME-VAR-L2  POP   DE                    Se aduce indicatorul 'noului'
                                            nume.
                EX    DE,HL                 Schimbare între registrii.
                INC   A                     Fanionul zero va fi setat
                                            dacă va fi o 'înlocuire'; va
                                            fi resetat în cazul unei
                                            'adăugări'.
                SCF                         Semnalare 'tratare variabile'
                CALL  092C,ME-ENTER         Acum se face intrarea.
                JR    08F0,ME-VAR-LP        Se ciclează bucla pentru a
                                            considera următoarea nouă
                                            variabilă.
```

THE 'MERGE A LINE OR A VARIABILE' SUBROUTINE (SUBRUTINA 'UNIRE LINIE SAU VARIABILA)

Această subrutină este introdusă cu următorii parametrii:

Fanionul de transport	resetat	- unire linie BASIC
	setat	- unire variabilă
Fanionul zero	resetat	- va fi o 'adăugare'
	setat	- va fi o 'înlocuire'
Registrul pereche HL		- punctul de început al unei noi intrări
Registrul pereche DE		- punctul în care se va face unirea

```
092C ME-ENTER     JR    NZ,093E,ME-ENT-1     Salt dacă se tratează o
                                             'adăugare'.
                  EX    AF,A'F'              Se salvează fanioanele.
                  LD    (X-PTR),HL           Se salvează indicatorul 'nou'
                  EX    DE,HL                până cînd vechea linie sau
                  CALL  1988,NEXT-ONE        variabilă se reface.
                  CALL  19E8,RECLAIM-2
                  EX    DE,HL
                  LD    HL,(X-PTR)
                  EX    AF,A'F'              Readucere fanioane.
```

Acum se poate face o nouă intrare.

```
093E ME-ENT-1     EX    AF,A'F'              Se salvează fanioanele.
                  PUSH  DE                   Se face o copie a
                                             indicatorului 'destinatie'.
                  CALL  1988,NEXT-ONE        Găseste lungimea 'noii'
                                             variabile/linii.
                  LD    (X-PTR),HL           Se salvează indicatorul
                                             'noii' variabile/linii.
                  LD    HL,(PROG)            Se aduce PROG - pentru a
                                             evita deteriorarea.
                  EX    (SP),HL              Se salvează PROG în stivă si
                                             se aduce 'noul' indicator.
                  PUSH  BC                   Se salvează lungimea.
                  EX    AF,A'F'              Se refac fanioanele.
                  JR    C,0955,ME-ENT-2      Salt înainte dacă se adaugă
                                             o nouă variabilă.
                  DEC   HL                   Se adaugă o nouă linie
                                             înaintea locatiei
                                             'destinatie'.
                  CALL  1655,MAKE-ROOM       Se pregăteste locatia pentru
                                             noua linie.
                  INC   HL
                  JR    0958,ME-ENT-3        Salt înainte.
0955 ME-ENT-2     CALL  1655,MAKE-ROOM       Se pregăteste locatia pentru
                                             noua variabilă.
0958 ME-ENT-3     INC   HL                   Se adresează prima locatie.
                  POP   BC                   Se reface lungimea.
                  POP   DE                   Regăseste PROG si îl
                  LD    (PROG),DE            păstrează la locul adecvat.
                  LD    DE,(X-PTR)           De asemenea, se aduce 'noul'
                                             indicator.
                  PUSH  BC                   Se salvează din nou lungimea
                  PUSH  DE                   si 'noul' pointer.
                  EX    DE,HL                Se schimbă indicatorii si se
                  LDIR                       copiază 'noua' variabilă /
                                             linie în locatia rezervată.
```

'Noua' variabilă/linie trebuie acum mutată din spatiul de lucru.

```
                  POP   HL                   Se aduce indicatorul 'nou'.
                  POP   BC                   Se aduce lungimea.
                  PUSH  DE                   Se salvează indicatorul
                                             'vechi'. (Se adresează
                                             locatia după 'adunarea'
                                             variabilei/liniei.)
                  CALL  19E8,RECLAIM-2       Se mută variabila/linia din
                                             spatiul de lucru.
                  POP   DE                   Revenire cu indicatorul
                  RET                        'vechi' în registrul pereche
                                             DE.
```

THE 'SAVE' CONTROL ROUTINE (RUTINA DE CONTROL SALVARE)
Operatia de salvare (SAVE) a programului sau a unui bloc de informatii este foarte utilizată.

```
0970 SA-CONTRL    PUSH  HL                   Se salvează 'indicatorul'.
                  LD    A,#FD                Se asigură că este deschis
                  CALL  1601,CHAN-OPEN       canalul K.
                  XOR   A                    Se semnalează 'primul mesaj'.
                  LD    DE,#09A1             Se tipăreste mesajul - Se
                  CALL  0C0A,PO-MSG          porneste banda, apoi se apasă
                                             oricare tastă.
                  SET   5,(IY-FLAG)          Semnalizare 'ecranul va cere
                                             să fie sters'.
                  CALL  15D4,WAIT-KEY        Se asteaptă apăsarea unei
                                             taste.
```

Atîta timp cît o tastă este apăsată, 'header'-ul este salvat.

```
                PUSH  IX                  Se salvează adresa de bază a
                                          'header'-ului în stiva
                                          calculatorului.
                LD    DE,+0011            Urmează să fie salvati 17
                                          octeti.
                XOR   A                   Se semnalează 'este un
                                          header'.
                CALL  04C2,SA-BYTES       Se trimite 'header'-ul; el
                                          este precedat de un octet
                                          'tip' si urmat de un octet
                                          'paritate'.
```

Acum urmează o scurtă întîrziere înaintea salvării programului / blocului de informatii.

```
                POP   IX                  Revenire indicator în
                                          'header'.
                LD    B,+32               Intîrzierea este pentru 50
0991 SA-1-SEC   HALT                      de întreruperi, aceasta
                DNZ   0991,SA-1-SEC       reprezentînd o secundă.
                LD    E,(IX+0B)           Se aduce lungimea blocului de
                LD    D,(IX+0C)           informatii care trebuie
                                          salvat.
                LD    A,+FF               Se semnalează 'bloc de
                                          informatii'.
                POP   IX                  Se aduce 'indicatorul de
                JP    04C2,SA-BYTES       început al blocului' si se
                                          salvează blocul.
```

THE CASSETTE MESSAGES (MESAJELE CASETEI)
Fiecare mesaj este dat cu ultimul caracter invertit (+80 hex.).

```
09A1 DEFB +80     - Octetul initial este sărit.
09A2 DEFM         - Pornire înregistrare, urmată de apăsarea oricărei
                    taste.
09C1 DEFM         - 'carriage return' - Program:
09CB DEFM         - 'carriage return' - Zonă număr:
09DA DEFM         - 'carriage return' - Zonă caracter:
09EC DEFM         - 'carriage return' - Octeti.
```

THE SCREEN PRINTER HANDLING ROUTINES (TRATAREA RUTINELOR DE AFISARE PE ECRAN)

THE 'PRINT - OUT' ROUTINES
Toate afisările din partea principală a ecranului, din partea de jos a ecranului si afisarea sînt tratate de acest set de rutine.
Rutina PRINT-OUT este introdusă cu registrul A, conținînd codul pentru controlul caracterului, caracterul bun de afisat sau un simbol.

```
09F4 PRINT-OUT    CALL  0B03,PO-FETC      Pozitia actuală de afisat.
                  CP    +20               Salt dacă codul reprezintă
                  JP    NC,0AD9,PO-ABLE   un caracter afisabil.
                  CP    +06               Se afisează semn de întrebare
                  JR    C,0A69,PO-QUEST   pentru coduri din intervalul
                                          +00 - +05.
                  CP    +18               Si de asemenea pentru coduri
                                          din intervalul +18 - +1F.
                  JR    NC,0A69,PO-QUEST
                  LD    HL,+0A01          Baza tabelului 'control'.
                  LD    E,A               Mutare cod în registrul
                  LD    D,+00             pereche DE.
                  ADD   HL,DE             Indexare în tabel si se aduce
                  LD    E,(HL)            deplasamentul.
                  ADD   HL,DE             Adunare deplasament si salt
                  PUSH  HL                indirect la subrutina
                  JP    0B03,PO-FETCH     adecvată.
```

THE 'CONTROL CHARACTER' TABLE (TABELUL 'CARACTERELOR DE CONTROL')

adresa	deplasament	caracter	adresa	deplasament	caracter
0A11	4E	afisare virgulă	0A1A	4F	nefolosit
0A12	57	EDIT	0A1B	5F	control cerneală
0A13	10	cursor stînga	0A1C	5E	control hîrtie
0A14	29	cursor dreapta	0A1D	5D	control lumină
0A15	54	cursor jos	0A1E	5C	control strălucire
0A16	53	cursor sus	0A1F	5B	control invers
0A17	52	DELETE (sterge)	0A20	5A	supracontrol
0A18	37	ENTER	0A21	54	control AT
0A19	50	nefolosit	0A22	53	control TAB

THE 'CURSOR LEFT' SUBROUTINE (SUBRUTINA 'CURSOR STINGA')
Subrutina este introdusă prin registrul B, care contine numărul liniei actuale, si prin registrul C, care contine numărul coloanei următoare.

```
0A23 PO-BACK-1    INC   C                 Mutare la stînga cu o
                                          coloană.
                  LD    A,+22             Se acceptă schimbarea numai
                  CP    C                 în partea stîngă.
                  JR    NZ,0A3A,PO-BACK-3
                  BIT   1,(FLAGS)         Dacă foloseste afisarea,
                  JR    NZ,0A38,PO-BACK-2 atunci salt înainte.
                  INC   B                 Ridicare cu o linie.
                  LD    C,+02             Setare valoare coloană.
                  LD    A,+18             Testare fată de linia maximă.
                  CP    B                 Notă: Aceasta trebuie să fie
                                          +19.
                  JR    NZ,0A3A,PO-BACK-3 Se acceptă schimbul doar în
                                          partea de sus a ecranului.
                  DEC   B                 Nu s-a acceptat, se coboară
                                          o linie.
0A38 PO-BACK-2    LD    C,+21             Setare coloană în partea
                                          stîngă.
0A3A PO-BACK-3    JP    0DD9,CL-SET       Revenire indirectă prin
                                          CL-SET & PO-STORE.
```

THE 'CURSOR RIGHT' SUBROUTINE (SUBRUTINA 'CURSOR DREAPTA')
Această subrutină execută o operatie identică formulării BASIC - PRINT OVER 1;CHR$ 32;-.

```
0A3D PO-RIGHT     LD    A,(P-FLAG)        Se aduce P-FLAG si se
                  PUSH  AF                salvează în stivă.
                  LD    (P-FLAG),+01      Setare P-FLAG la OVER 1.
                  LD    A,+20             In A se încarcă 'spatiu'.
                  CALL  0B65,PO-CHAR      Afisare caracter.
                  POP   AF                Se aduce vechea valoare a
                  LD    (P-FLAG),A        lui P-FLAG.
                  RET                     Terminare.
                                          Notă: Programatorul a uitat
```

să iasă prin PO-STORE.

THE 'CARRIAGE RETURN' SUBROUTINE (SUBRUTINA 'CARRIAGE RETURN')
Dacă tipărirea se face pe ecran (afisare), atunci se face un test pentru defilare pe ecran ('scroll?') înainte de a decrementa numărul liniei.

```
0A4F PO-ENTER    BIT   1,(FLAGS)          Salt înainte dacă se tratează
                 JP    NZ,0ECB,COPY-BUFF  tipărire.
                 LD    C,+21              Setare coloană în partea
                                          stîngă.
                 CALL  0C55,PO-SCR        Defilare, dacă este necesar.
                 DEC   B                  Acum se coboară o linie.
                 JP    0BB9,CL-SET        Se execută acum o revenire
                                          indirectă, prin CL-SET &
                                          PO-STORE.
```

THE 'PRINT COMMA' SUBROUTINE (SUBRUTINA 'AFISARE VIRGULA')
Valoarea coloanei actuale este schimbată și registrul A este setat cu +00 (pentru TAB 0) sau +10 (pentru TAB 16).

```
0A5F PO-COMMA    CALL  0B03,PO-FETCH      De ce se reia?
                 LD    A,C                Numărul coloanei actuale.
                 DEC   A                  Se mută în partea dreaptă cu
                 DEC   A                  două coloane si apoi se
                                          testează.
                 AND   +10                Registrul A va conține +00
                                          sau +10.
                 JR    0AC3,PO-FILL       Se iasă prin PO-FILL.
```

THE 'PRINT A QUESTION MARK' SUBROUTINE (SUBRUTINA 'AFISAREA UNUI SEMN DE INTREBARE')
Un semn de întrebare se afișează întotdeauna cînd se face o încercare de afisare a unui cod eronat.

```
0A69 PO-QUEST    LD    A,+3F              Caracterul '?'.
                 JR    0AD9,PO-ABLE       Se afișează acest caracter în
                                          locul celui nevalidat pentru
                                          afisare.
```

THE 'CONTROL CHARACTERS WITH OPERANDS' ROUTINE (RUTINA 'CONTROLUL CARACTERELOR CU OPERANZI')
Controlul caracterelor de la INK (cerneală) la OVER necesită un singur operand, în timp ce controlul caracterelor AT &TAB necesită să fie urmate de doi operanzi.
Această rutină duce la salvarea codului de control al caracterului în TVDATA-lo, primul operand în TVDATA-hi sau în registrul A dacă este cerut un singur operand, și al doilea operand în registrul A.

```
0A6D PO-TV-2     LD    DE,+0A87           Salvarea primului operand în
                 LD    (TVDATA-hi),A      TVDATA-hi si schimbarea
                 JR    0A80,PO-CHANGE     adresei rutinei de 'iesire'
                                          în PO-CONT (+0A87).
```

Aici se va intra cînd se tratează caracterele AT & TAB.

```
0A75 PO-2-OPER   LD    DE,+0A6D           Codul caracterului se
                 JR    0A7D,PO-TV-1       salvează în TVDATA-lo si
                                          adresa rutinei de 'iesire'
                                          se schimbă în PO-TV-2
                                          (+0A6D).
```

Aici se va intra cînd se tratează fiecare culoare - INK la OVER.

```
0A7A PO-1-OPER   LD    DE,+0A87           Rutina 'iesire' va fi
                                          schimbată în PO-CONT (+0A87).
0A7D PO-TV-1     LD    (TVDATA-lo),A      Salvare cod caracter control.
```

Adresa rutinei curente de 'iesire' este schimbată pentru un timp.

```
0A80 PO-CHANGE   LD    HL,(CURCHL)        Registrul HL va conține
                                          adresa rutinei 'iesire'.
                 LD    (HL),E             Se introduce noua adresa a
                 INC   HL                 rutinei 'iesire', fortînd
                 LD    (HL),D             astfel ca următorul cod de
                 RET                      caracter să fie considerat
                                          operand.
```

Odată ce operanzii au fost adunati (strînsi) rutina continuă.

```
0A87 PO-CONT    LD    DE,+09F4          Se rememorează adresa
                CALL  0A80,PO-CHANGE    originală pentru PRINT-OUT
                                        (+09F4).
                LD    HL,(TVDATA)       Se aduce codul de control si
                                        primul operand dacă sînt doi
                                        operanzi.
                LD    D,A               'Ultimul' operand si codul de
                LD    A,L               control sînt mutate.
                CP    +16               Salt înainte dacă se tratează
                JP    C,2211,CO-TEMPS   INK to OVER.
                JR    NZ,0AC2,PO-TAB    Salt înainte dacă se tratează
                                        TAB.
```

Acum se lucrează cu caracterul de control AT.

```
                LD    B,H               Numărul liniei.
                LD    C,D               Numărul coloanei.
                LD    A,+1F             Se schimbă numărul coloanei,
                SUB   C                 astfel încît +00 - +1F devine
                                        +1F - +00.
                JR    C,0AAC,PO-AT-ERR  Trebuie să fie în ordine.
                ADD   A,+02             Se adună la deplasament
                LD    C,A               pentru a pune în C +21 - +22.
                BIT   1,(FLAGS)         Salt înainte dacă se tratează
                JR    NZ,0ABF,PO-AT-SET afisare.
                LD    A,+16             Se schimbă numărul liniei,
                SUB   B                 astfel încît +00 - +15 devine
                                        +16 - +01.
0AAC PO-AT-ERR  JP    C,1E9F,REPORT-B   Dacă conditia este
                                        îndeplinită, salt înainte.
                INC   A                 Sirul +16 - +01 devine +17 -
                LD    B,A               +02.
                INC   B                 Si acum +18 - +03.
                BIT   0,(TV-FLAG)       Dacă afisarea se face în
                JP    NZ,0C55,PO-SCR    partea de jos a ecranului se
                                        consideră dacă defilarea este
                                        necesară.
                CP    (DF-SZ)           Se dă prezentarea 5 - afară
                JP    C,0C86,REPORT-5   din ecran, dacă se cere.
0ABF PO-AT-SET  JP    0DD9,CL-SET       Revenire prin CL-SET&PO-STORE
```

Si cu caracterul de control TAB.

```
0AC2 PO-TAB     LD    A,H               Se aduce primul operand.
0AC3 PO-FILL    CALL  0B03,PO-FETCH     Pozitia de afisare actuală.
                ADD   A,C               Se adună valoarea coloanei
                                        curente.
                DEC   A                 Găseste cîte spatii modulo
                AND   +1F               32 sînt cerute si revenire
                RET   Z                 dacă rezultatul este zero.
                LD    D,A               Se foloseste D ca si contor.
                SET   0,(FLAGS)         Se 'suprapun' spatiile de
                                        fond.
0AD0 PO-SPACE   LD    A,+20             Se afiseză 'numărul D' de
                CALL  0C3B,PO-SAVE      spatii.
                DEC   D
                JR    NZ,0AD0,PO-SPACE
                RET                     Acum se termină.
```

PRINT THE CHARACTER CODES (AFISAREA CODULUI CARACTERULUI)
Caracterul (sau caracterele) cerute sînt afisate prin apelarea PO-ANY urmat de PO-STORE.

```
0AD9 PO-ABLE    CALL  0B24,PO-ANY       Se tipăreste caracterul
                                        (caracterele) si se continuă
                                        în PO-STORE.
```

THE 'POSITION STORE' SUBROUTINE (SUBRUTINA 'REZERVARE POZITIE')
Noua valoare a poziției 'linie & coloană' si adresa 'pixelului' sînt stocate în sistemul propriu de variabile.

```
0ADC PO-STORE   BIT   1,(FLAGS)         Salt înainte pentru tratare
                JR    NZ,0AFC,PO-ST-PR  afisare.
                BIT   0,(TV-FLAG)       Salt înainte dacă se tratează
                JR    NZ,0AF0,PO-ST-E   partea de jos a ecranului.
                LD    (S-POSN),BC       Se salvează valorile care
                LD    (DF-CC),HL        descriu partea principală a
                RET                     ecranului. Apoi revenire.
0AF0 PO-ST-E    LD    (S-POSNL),BC      Se salvează valorile care
```

```
                LD    (ECHO-E),BC       descriu partea de jos a
                LD    (DF-CCL),HL       ecranului.
                RET                     Apoi revenire.
0AFC PO-ST-PR   LD    (P-POSN),C        Se salveazà valorile care
                LD    (PR-CC),HL        descriu bufferul de afisare.
                RET                     Apoi revenire.
```

THE 'POSITION FETCH' SUBROUTINE (SUBRUTINA 'ADUCERE POZITIE')
Parametrii actuali ai pozitiei sînt adusi din sistemul propriu de variabile.

```
0B03 PO-FETCH   BIT   1,(FLAGS)         Salt înainte dacà se tratează
                JR    NZ,0B1D,PO-F-PR   afisare.
                LD    BC,(S-POSN)       Se aduc valorile ce descriu
                LD    HL,(DF-CC)        partea principalà a ecranului
                BIT   0,(TV-FLAG)       si revenire dacà aceasta a
                RET   Z                 fost intentia.
                LD    BC,(S-POSNL)      Altfel, se aduc valori care
                LD    HL,(PR-CC)        descriu partea de jos a
                RET                     ecranului.
0B1D PO-F-PR    LD    C,(P-POSN)        Se aduc valorile care descriu
                LD    HL,(PR-CC)        bufferul de afisare.
                RET
```

THE 'PRINT ANY CHARACTER(S)' SUBROUTINE (SUBRUTINA 'AFISARE ORICARUI CARACTER (CARACTERE)')
Codurile caracterelor actuale, codurile simbolurilor, codurile grafice definite de utilizator si codurile grafice sînt prelucrate separat.

```
0B24 PO-ANY     CP    +80               Salt înainte pentru cod
                JR    C,0B65,PO-CHAR    caracter actual.
                CP    +90               Salt înainte pentru cod semn
                JR    NC,0B52,PO-T&UDG  sau cod semn grafic definit
                                        de utilizator (UDG).
                LD    B,A               Transfer cod grafic.
                CALL  0B38,PO-GR-1      Formare formà graficà.
                CALL  0B03,PO-FETCH     HL a fost modificat, deci se
                                        'reface'.
                LD    DE,+5C92          Se încarcà în DE începutul
                                        formei grafice; aceasta este
                                        MEMBOT.
                JR    0B7F,PO-ALL       Salt înainte pentru tipàrire
                                        cod grafic.
```

Caracterele grafice sînt construite într-o manierà Ad Hoc în spatiul de memorie al calculatorului; acestea sînt MEM-0 & MEM-1.

```
0B38 PO-GR-1    LD    HL,+5C92          Incarcà MEMBOT.
                CALL  0B3E,PO-GR-2      Ca urmare, se apeleazà
                                        subrutina urmàtoare de douà
                                        ori.
0B3E PO-GR-2    RR    B                 Se determinà bitul 0 (si mai
                SBC   A,A               tîrziu bitul 2) al codului
                                        grafic.
                AND   +0F               Registrul A va contine +00
                                        sau +0F, în functie de
                                        valoarea bitului în cadrul
                                        codului.
                LD    C,A               Se salveazà rezultatul în C.
                RR    B                 Se determinà bitul 1 (si mai
                SBC   A,A               tîrziu bitul 3) al codului
                                        grafic.
                AND   +F0               Registrul A va contine +00
                                        sau +F0.
                OR    C                 Cele douà rezultate sînt
                                        combinate.
                LD    C,+04             Registrul A va contine o
0B4C PO-GR-3    LD    (HL),A            jumàtate a caracterului
                INC   HL                format, ce va fi utilizat de
                                        patru ori.
                DEC   C                 Aceasta se face pentru partea
                JR    NZ,0B4C,PO-GR-3   superioarà a caracterului si
                RET                     apoi pentru partea sa
                                        inferioarà.
```

Codurile seanelor si codurile seanelor grafice definite de càtre utilizator sînt acum separate.

```
0B52 PO-T&UDG   SUB   +A5                Salt înainte cu codul grafic.
```

```
                JR    NC,0B5F,PO-T
                ADD   A,+15              Codurile UDG sînt acum +00 -
                                         +0F.
                PUSH  BC                 Salvarea valorii pozitiei
                                         actuale în stiva
                                         calculatorului.
                LD    BC,(UDG)           Se aduce adresa de bază a
                JR    0B6A,PO-CHAR-2     zonei UDG si salt înainte.
0B5F PO-T       CALL  0C10,PO-TOKENS     Acum se face afisare semn si
                JP    0B03,PO-FETCH      revenire prin PO-FETCH.
```

Este identificată forma caracterului cerut.

```
0B65 PO-CHAR    PUSH  BC                 Este salvată pozitia curentă.
                LD    BC,(CHARS)         Se aduce adresa de bază a
                                         spatiului caracterului
0B6A PO-CHAR-2  EX    DE,HL              Este salvată adresa la care
                                         se afisează.
                LD    HL,+5C3B           Aceasta este FLAGS.
                RES   0,(HL)             Permite spatii de fond.
                CP    +20                Salt înainte dacă caracterul
                JR    NZ,0B76,PO-CHAR-3  nu este 'spatiu'.
                SET   0,(HL)             Dar daca este 'se suprapune'.
0B76 PO-CHAR-3  LD    H,+00              Acum se trece codul
                LD    L,A                caracterului în registrul
                                         pereche HL.
                ADD   HL,HL              De fapt, codul caracterului
                ADD   HL,HL              este inmultit cu 8.
                ADD   HL,HL
                ADD   HL,BC              Adresa de bază a formei
                                         caracterului este găsită.
                POP   BC                 Se aduce pozitia actuală si
                EX    DE,HL              adresa de bază se trece în
                                         registrul pereche HL.
```

THE 'PRINT ALL CHARACTERS' SUBROUTINE (SUBRUTINA 'TIPARIREA TUTUROR CARACTERELOR')
Această subrutină este folosită pentru afisarea tututor '8*8' biti caracter. La intrare, registrul pereche DE contine adresa bazei caracterului format, registrul pereche HL contine adresa destinatiei si registrul pereche BC contine valoarea curentă 'linie & coloană'.

```
0B7F PR-ALL     LD    A,C                Se aduce numărul coloanei.
                DEC   A                  Mutare cu o coloană la
                                         dreapta.
                LD    A,+21              Salt înainte doar dacă se
                JR    NZ,0B93,PR-ALL-1   indică o nouă linie.
                DEC   B                  Coborîre cu o linei.
                LD    C,A                Numărul coloanei este +21.
                BIT   1,(FLAGS)          Salt înainte dacă se tratează
                JR    Z,0B93,PR-ALL-1    ecranul.
                PUSH  DE                 Salvarea adresei de bază pînă
                CALL  0ECD,COPY-BUFF     cînd se goleste bufferul de
                POP   DE                 afisare.
                LD    A,C                Se copiază noul număr al
                                         coloanei.
0B93 PR-ALL-1   CP    C                  Testare dacă o linie nouă
                PUSH  DE                 urmează să fie folosită. Dacă
                CALL  Z,0C55,PO-SCR      da, se cercetează dacă
                POP   DE                 display-ul cere defilare.
```

Se prezintă acum starea 'INVERSE & OVER'.

```
                PUSH  BC                 Se salvează valoarea pozitiei
                PUSH  HL                 si a adresei destinatie în
                                         stiva calculatorului.
                LD    A,(P-FLAG)         Se aduce P-FLAG si se citeste
                                         bitul 0.
                LD    B,+FF              Se pregăteste 'OVER-mask' în
                RRA                      registrul B; aceasta este
                JR    C,0BA4,PR-ALL-2    OVER 0 = +00 & OVER 1 = +FF.
                INC   B
0BA4 PR-ALL-2   RRA                      Se citeste bitul 2 din P-FLAG
                RRA                      si se pregăteste 'INVERSE-
                SBC   A,A                mask' în registrul C; aceasta
                LD    C,A                este INVERSE 0 = +00 &
                                         INVERSE 1 = +FF.
                LD    A,+08              Se pune registrul A în
                AND   A                  asteptarea contorului 'pixel-
```

```
                                          linie' si se sterge fanionul
                                          de transport.
                BIT   1,(FLAGS)           Salt înainte dacă se tratează
                JR    Z,0BB6,PR-ALL-3     ecranul.
                SET   1,(FLAGS2)          Se semnalează 'bufferul de
                                          afisare nu mai este gol'.
                SCF                       Se pune fanionul de transport
                                          să arate că s-a utilizat
                                          printer-ul.
0BB6 PR-ALL-3   EX    DE,HL               Se schimbă adresa de
                                          destinatie cu adresa de bază
                                          înainte de a intra în buclă.
```

Caracterul poate fi acum tipărit. Se fac opt treceri prin buclă - una pentru fiecare 'linie-pixel'.

```
0BB7 PR-ALL-4   EX    AF,A'F'             Fanionul de transport este
                                          setat când se utilizează
                                          afisarea. Acest flag se
                                          salvează în F'.
                LD    A,(DE)              Se aduce 'linia - pixel'
                                          existentă.
                AND   B                   Se foloseste 'OVER-mask' si
                XOR   (HL)                se compară rezultatul cu
                                          'linia-pixel' a caracterului
                                          format.
                XOR   C                   In final se consideră
                                          'INVERS-mask'.
                LD    (DE),A              Se introduce rezultatul.
                EX    AF,A'F'             Se aduce fanionul de
                JR    C,0BB3,PR-ALL-6     imprimare si salt înainte,
                                          dacă este cerut.
                INC   D                   Se incrementează adresa
                                          destinatie.
0BC1 PR-ALL-5   INC   HL                  Se incrementează 'linia -
                                          pixel' a caracterului format.
                DEC   A                   Se decrementează contorul si
                JR    NZ,0BB7,PR-ALL-4    se bucleză înapoi numai dacă
                                          este zero.
```

Odată ce s-a afisat caracterul, octetul atribut se setează la cerere.

```
                EX    DE,HL               Registrul H va conține adresa
                DEC   H                   superioară corectă pentru
                                          zona caracterului.
                BIT   1,(FLAGS)           Se setează octetul atribut
                CALL  Z,0BDB,PO-ATTR      numai dacă se tratează
                                          ecranul.
                POP   HL                  Se reface adresa destinatie
                POP   BC                  initială si valorile
                                          pozitiei.
                DEC   C                   Se decrementează numărul
                INC   HL                  coloanei si se incrementează
                RET                       adresa destinatie înainte
                                          de revenire.
```

Când se foloseste imprimanta, adresa destinatie trebuie ridicată prin incrementare cu +20.

```
0BB3 PO-ALL-6   EX    AF,A'F'             Salvare fanion afisare.
                LD    A,+20               Se încarcă valoarea cerută
                                          pentru incrementare.
                ADD   A,E                 Se adună valoarea si
                LD    E,A                 rezultatul se pune înapoi
                                          în registrul E.
                EX    AF,A'F'             Se aduce fanionul.
                JR    0BC1,PR-ALL-5       Salt înapoi în buclă.
```

THE 'SET ATTRIBUTE BYTE' SUBROUTINE (SUBRUTINA 'SETARE OCTET ATRIBUT')
Octetul atribut adecvat este identificat si încărcat. Noua valoare este formată prin modificarea valorii vechi, ATTR-T, MASK-T si P-FLAG. In final noua valoare este copiată în spatiul atribuit.

```
0BDB PO-ATTR    LD    A,H                 Octetul superior al adresei
                RRCA                      destinatie este divizat cu 8
                RRCA                      si adunat cu +03 pentru a
                RRCA                      determina care treime de
                AND   +03                 ecran este adresată; aceasta
```

```
                        OR    +58           este 00, 01 sau 02.
                        LD    H,A           Octetul superior pentru
                                            spatiul atribuit este acum
                                            format.
                        LD    DE,(ATTR-T)   D contine ATTR-T si E contin
                                            MASK-T.
                        LD    A,(HL)        Vechea valoare
                                            caracteristică.
                        XOR   E             Valorile lui ATTR-T si MASK-
                        AND   D             sînt luate în calcul.
                        XOR   E
                        BIT   6,(P-FLAG)    Salt înainte doar dacă se
                        JR    Z,0BFA,PO-ATTR-1 lucrează cu PAPER 9.
                        AND   +C7           Vechea culoare a hîrtiei este
                                            ignorată, si după cum
                        BIT   2,A           culoarea cernelii este
                        JR    NZ,0BFA,PO-ATTR-1 luminoasă sau întunecată,
                        XOR   +38           noua culoare a hîrtiei va fi
                                            negru (000) sau alb (111).
0BFA PO-ATTR-1          BIT   4,(P-FLAG)    Salt înainte numai dacă se
                        JR    Z,0C08,PO-ATTR-2 lucrează cu INK 9.
                        AND   +F8           Vechea culoare de cerneală
                                            este ignorată, si după cum
                        BIT   5,A           culoarea hîrtiei este
                        JR    NZ,0C08,PO-ATTR-2 întunecată sau luminoasă,
                        XOR   +07           cerneala va avea culoarea
                                            neagră (000) sau albă (111).
0C08 PO-ATTR-2          LD    (HL),A        Se încarcă noua valoare
                        RET                 caracteristică si se revine.
```

THE 'MESSAGE PRINTING' SUBROUTINE (SUBRUTINA 'TIPARIRE MESAJ')
Această subrutină este folosită pentru tipărirea mesajelor si a semnelor.
Registrul A contine 'numărul intrării' mesajului sau semnului într-un tabel.
Registrul pereche DE contine adresa de bază a tabelului.

```
0C0A PO-MSG             PUSH  HL            Octetul superior al ultimei
                        LD    H,+00         intrări în stiva
                        EX    (SP),HL       calculatorului este pus pe
                                            zero, astfel încît să
                                            suprapună spatii (vezi mai
                                            jos).
                        JR    0C14,PO-TABLE Salt înainte.
```

Aici se intră dacă se însiră un cod de semn.

```
0C10 PO-TOKENS          LD    DE,+0095      Adresa de bază a tabelului
                                            de semn.
                        PUSH  AF            Se salvează codul în stivă.
                                            (Ordinea +00 - +5A;RND-COPY).
```

Se cercetează tabelul si intrarea corectă este afisată.

```
0C14 PO-TABLE           CALL  0C41,PO-SEARCH Localizare intrare cerută.
                        JR    C,0C22,PO-EACH Afisare mesaj/semn.
                        LD    A,+20         Se va afisa un 'spatiu'
                        BIT   0,(FLAGS)     înaintea mesajului/semnului
                        CALL  Z,0C3B,PO-SAVE cerut.
```

Caracterele mesajului/semnului sînt afisate unul după altul.

```
0C22 PO-EACH            LD    A,(DE)        Incărcare cod.
                        AND   +7F           Se anulează orice 'bit
                                            invertit'.
                        CALL  0C3B,PO-SAVE  Afisare caracter.
                        LD    A,(DE)        Se încarcă din nou codul.
                        INC   DE            Avans indicator.
                        ADD   A,A           'Bitul invertit' se încarcă
                        JR    NC,0C22,PO-EACH în fanionul de transport si
                                            semnalează sfîrsitul
                                            mesajului/semnului; în caz
                                            contrar, salt înapoi.
```

Acum se consideră dacă este necesar un 'spatiu următor'.

```
                        POP   DE            Pentru mesaje- D contine +00;
                                            pentru semne- D contine +00
                                            - +5A.
                        CP    +48           Salt înainte dacă ultimul
                        JR    Z,0C35,PO-TRSP caracter a fost '$'.
```

```
                CP    +82                 Revenire dacă ultimul
                RET   C                   caracter a fost oricare altul
                                          înainte de 'A'.
0C35 PO-TRSP    LD    A,D                 Se examinează valoarea din D
                CP    +03                 și revenire dacă se indică un
                RET   C                   mesaj, RND, INKEY$ sau PI.
                LD    A,+20               Toate celelalte cazuri vor
                                          cere un 'spatiu următor'.
```

THE 'PO-SAVE' SUBROUTINE (SUBRUTINA 'PO-SAVE')
Această subrutină permite scrierea 'recursivă' a caracterelor. Registrii necesari sînt salvați pînă cînd se apelează 'PRINT-OUT'.

```
0C3B PO-SAVE    PUSH  DE                  Se salvează registrul
                                          pereche DE.
                EXX                       Se salvează HL & BC.
                RST   0010,PRINT-A-1      Afișare caracter singular.
                EXX                       Refacere HL & BC.
                POP   DE                  Refacere DE.
                RET                       Terminare.
```

THE 'TABLE SEARCH' SUBROUTINE (SUBRUTINA 'CAUTARE TABEL')
Subrutina revine cu registrul pereche DE indicînd caracterul inițial al intrării cerute și fanionul de transport este resetat dacă trebuie considerat un 'spatiu de fond'.

```
0C41 PO-SEARCH  PUSH  AF                  Se salvează 'număr intrare'.
                EX    DE,HL               HL conține acum adresa de
                                          bază.
                INC   A                   Se face ordinul +01 ?
0C44 PO-STEP    BIT   7,(HL)              Se așteaptă un 'caracter'
                INC   HL                  invers.
                JR    Z,0C44,PO-STEP
                DEC   A                   Se caută intrările pînă se
                JR    NZ,0C44,PO-STEP     găsește cea corectă.
                EX    DE,HL               DE indică caracterul inițial.
                POP   AF                  Incărcare 'număr intrare' și
                CP    +20                 revenire cu transportul setat
                RET   C                   pentru primele 32 de intrări.
                LD    A,(DE)              Oricum, dacă caracterul
                SUB   +41                 inițial este o literă ar
                RET                       ar putea fi nevoie de un
                                          spatiu de fond.
```

THE 'TEST FOR SCROLL' SUBROUTINE (SUBRUTINA 'TEST PENTRU DEFILARE')
Această subrutină este apelată cînd este necesară defilarea ecranului. Aceasta se poate întîmpla în trei cazuri: i. cînd se tratează un caracter 'carriage return'; ii. cînd se folosește AT într-o linie INPUT; iii. cînd linia curentă este plină și trebuie folosită linia următoare.
La intrare registrul B va conține numărul liniei după test.

```
0C55 PO-SCR     BIT   1,(FLAGS)           Revenire imediată dacă
                RET   NZ                  afișarea a fost utilizată.
                LD    DE,+0DD9            Preîncărcarea stivei
                PUSH  DE                  calculatorului cu adresa lui
                                          'CL-SET'.
                LD    A,B                 Se transferă numărul liniei.
                BIT   0,(TV-FLAG)         Salt înainte dacă se
                JP    NZ,0B02,PO-SCR-4    consideră 'INPUT ... AT ...',
                CP    (DF-SZ)             revenire, prin CL-SET, dacă
                JR    C,0C86,REPORT-5     numărul liniei este mai mare
                RET   NZ                  decît valoarea lui DF-SZ; se
                                          dă prezentarea 5 dacă este
                                          mai mic; altfel se continuă.
                BIT   4,(TV-FLAG)         Salt înainte numai dacă se
                JR    Z,0C88,PO-SCR-2     lucrează cu 'listare
                                          automată'.
                LD    E,(BREG)            Se aduce numărătorul de
                                          linii.
                DEC   E                   Se decrementează numărătorul.
                JR    Z,0CD2,PO-SCR-3     Salt înainte dacă listarea
                                          trebuie rulată.
                LD    A,+00               Altfel se deschide canalul
                CALL  1601,CHAN-OPEN      'K', se reface indicatorul de
                LD    SP,(LIST-SP)        stivă, se repoziționează
                RES   4,(TV-FLAG)         fanionul dacă s-a terminat
                RET                       listarea automată și se
                                          revine prin CL-SET.
```

Prezentarea 5 - afară din ecran.

```
0C86 REPORT-5     RST   0008,ERROR-1      Se apelează rutina de tratare
                  DEFB  +04                erori.
```

Acum se ia în considerare dacă s-a cerut prompt 'scroll?' ('defilare'?).

```
0C88 PO-SCR-2     DEC   (SCR-CT)           Se decrementează contorul de
                  JR    NZ,0CD2,PO-SCR-3   rulare si se trece la redare
                                           numai dacă a ajuns pe zero.
```

Se trece la redarea unui mesaj.

```
                  LD    A,+18              Contorul este resetat.
                  SUB   B
                  LD    (SCR-CT),A
                  LD    HL,(ATTR-T)        Valorile curente ale lui
                  PUSH  HL                 ATTR-T si MASK-T sînt
                                           salvate.
                  LD    A,(P-FLAG)         Se salvează valoarea curentă
                  PUSH  AF                 a lui P-FLAG.
                  LD    A,+FD              Se deschide canalul 'K'.
                  CALL  0601,CHAN-OPEN
                  XOR   A                  Mesajul 'scroll?' este mesaj
                  LD    DE,+0CF8           '0'. Acest mesaj este acum
                  CALL  0C0A,PO-MSG        afisat.
                  SET   5,(TV-FLAG)        Semnal 'sterge partea de jos
                                           a ecranului după o apăsare de
                                           tastă'.
                  LD    HL,+5C3B           Acesta este FLAGS.
                  SET   3,(HL)             Semnal 'mod L'.
                  RES   5,(HL)             Semnal 'încă nu este nici o
                                           tastă'.
                  EXX                      Notă: Si DE trebuie salvat în
                                           stivă.
                  CALL  15D4,WAIT-KEY      Se încarcă codul unei singure
                                           taste.
                  EXX                      Se refac registrii.
                  CP    +20                Se face un salt înainte la
                  JR    Z,0D00,REPORT-D    REPORT-D - BREAK-CONT
                  CP    +E2                repetare' - dacă s-a apăsat
                  JR    Z,0D00,REPORT-D    tasta 'BREAK', 'STOP', 'N'
                  OR    +20                sau 'n'; altfel, se acceptă
                  CP    +6E                că apăsarea tastei a indicat
                  JR    Z,0D00,REPORT-D    necesitatea rulării ecranului
                  LD    A,+FE              Se deschide canalul 'S'.
                  CALL  1601,CHAN-OPEN
                  POP   AF                 Se refac valoarea din
                  LD    (P-FLAG),A         P-FLAG.
                  POP   HL                 Se refac valorile lui ATTR-T
                  LD    (ATTR-T),HL        si MASK-T.
```

Ecranul este acum rulat.

```
0CD2 PO-SCR-3     CALL  0DFE,CL-SC-ALL     Intregul ecran este rulat.
                  LD    B,(DF-SZ)          Numerele de linie si coloană
                  INC   B                  pentru începutul liniei
                  LD    C,+21              dinaintea părtii de jos a
                  PUSH  BC                 ecranului sînt găsite si
                                           salvate.
                  CALL  0E9B,CL-ADDR       Octetul atribut corespunzător
                  LD    A,H                pentru această portiune de
                  RRCA                     caracter este găsit.
                  RRCA                     Registrul pereche HL contine
                  RRCA                     adresa octetului.
                  AND   +03
                  OR    +58
                  LD    H,A
```

Linia în discutie va avea valoarea caracteristică a părtii inferioare' si noua linie din partea de jos a ecranului poate avea valoarea 'ATTR-P', asa că valorile caracteristice sînt schimbate.

```
                  LD    DE,+5AE0           DE contine primul octet
                                           atribut al liniei inferioare.
                  LD    A,(DE)             Valoarea este încărcată.
                  LD    C,(HL)             Valoarea 'partea inferioară'.
                  LD    B,+20              Acolo sînt 32 de octeti.
                  EX    DE,HL              Schimbare indicatori.
```

```
0CF0 PO-SCR-3A   LD    (DE),A          Se face prima schimbare si
                 LD    (HL),C          apoi se trece la utilizarea
                 INC   DE              aceleasi valori pentru cei 32
                 INC   HL              de octeti atribut pentru
                 DJNZ  0CF0,PO-SCR-3A  tratarea a două linii.
                 POP   BC              Se încarcă numerele de linie
                                       si coloană ale liniei
                                       inferioare a 'părtii de sus'
                 RET                   înainte de a se reveni.
```

Mesajul 'rulare'.

```
0CF8             DEFB  +80             Marcatorul initial - pas
                                       înainte.
                 DEFB  +73,+63,+72,+6F s - c - r - o
                 DEFB  +6C,+6C,+BF     l - l - ? (convertite).
```

Prezentarea D- BREAK - CONT se repetă.

```
0D00 REPORT-D    RST   0008,ERROR-1    Se apelează rutina de tratare
                 DEFB  +0C             a erorilor.
```

Partea inferioară a ecranului este prezentată în continuare:

```
0D02 PO-SCR-4    CP    +02             Eroarea 'iesire din ecran'
                 JR    C,0C86,REPORT-5 se dă dacă partea inferioară
                 ADD   A,(DF-SZ)       urmează să fie 'prea mare' si
                 SUB   +19             se revine dacă rularea nu
                 RET   NC              este necesară.
                 NEG                   Registrul A va contine acum
                                       'numărul rulărilor ce trebuie
                                       făcute'.
                 PUSH  BC              Numerele de linie si coloană
                                       sînt acum salvate.
                 LD    B,A             Acum se salvează 'numărul
                 LD    HL,(ATTR-T)     rulărilor', ATTR-T, MASK-T &
                 PUSH  HL              P-FLAG.
                 LD    HL,(P-FLAG)
                 PUSH  HL
                 CALL  0D4D,TEMPS      Trebuie folosite numerele
                                       culorilor 'permanente'.
                 LD    A,B             Încărcare 'număr rulări'.
```

Partea inferioară a ecranului este acum rulată de un număr 'A' de ori.

```
0D1C PO-SCR-4A   PUSH  AF              Salvare 'număr'.
                 LD    HL,+5C6B        Acesta este DF-SZ.
                 LD    B,(HL)          Valoarea din DF-SZ este
                 LD    A,B             incrementată; registrul B va
                 INC   A               contine valoarea formată iar
                 LD    (HL),A          registrul A noua valoare.
                 LD    HL,+5C89        Încărcare S-POSN-hi.
                 CP    (HL)            Saltul are loc numai dacă
                 JR    C,0D2D,PO-SCR-4B partea inferioară a ecranului
                                       urmează a fi rulată (B=vechea
                                       DF-SZ).
                 INC   (HL)            Altfel, S-POSN-hi este
                 LD    B,+18           incrementat si întregul ecran
                                       este rulat (B=+18).
0D2D PO-SCR-4B   CALL  0E00,CL-SCROLL  Rulare 'B' linii.
                 POP   AF              Se aduce si se decrementează
                 DEC   A               'numărul de rulare'.
                 JR    NZ,0D1C,PO-SCR-4A Salt înapoi pînă la terminare
                 POP   HL              Se refăce valoarea lui P-FLAG
                 LD    (P-FLAG),L
                 POP   HL              Se refac valorile lui ATTR-T
                 LD    (ATTR-T),HL     si MASK-T.
                 LD    BC,(S-POSN)     Dacă S-POSN a fost schimbat,
                 RES   0,(TV-FLAG)     este apelat CL-SET pentru a-i
                 CALL  0DD9,CL-SET     egala valoarea cu DF-CC.
                 SET   0,(TV-FLAG)     Se resetează fanionul pentru
                 POP   BC              ca să indice că s-a tratat
                 RET                   partea inferioară a ecranului
                                       se aduc numerele liniei si
                                       coloanei si se revine.
```

THE 'TEMPORARY COLOUR ITEMS' SUBROUTINE (SUBRUTINA 'NUMERELE DE CULORI TEMPORARE')

Aceasta este o rutină foarte importantă. Ea este utilizată întotdeauna cînd

trebuie copiate detaliile 'permanente' în sistemul 'temporar' de variabile. Se consideră mai întîi ATTR-T &MASK-T.

```
0D4D TEMPS        XOR   A              Registrul A se setează (+00).
                  LD    HL,(ATTR-P)    Se încarcă valorile curente
                  BIT   0,(IY-FLAG)    ale lui ATTR-T si MASK-P.
                  JP    Z,0D5B,TEMPS-1 Salt înainte dacă se tratează
                                       partea principală a
                                       ecranului.
                  LD    H,A            Altfel se utilizează +00 si
                  LD    L,(BORDCR)     valoarea din BORDCR n loc.
0D5B TEMPS-1      LD    (ATTR-T),HL    Setare ATTR-T & MASK-T.
```

In continuare se consideră P-FLAG.

```
                  LD    HL,+5C91       Acesta este P-FLAG.
                  JR    NZ,0D65,TEMPS-2 Salt înainte dacă se lucrează
                                       cu partea inferioară a
                                       ecranului (A = +00).
                  LD    A,(HL)         Altfel se încarcă valoarea
                  RRCA                 P-FLAG si se mută bitii
                                       impari în locul celor pari.
0D65 TEMPS-2      XOR   (HL)           Se trece la copierea bitilor
                  AND   +55            pari din A în P-Flag.
                  XOR   (HL)
                  LD    (HL),A
                  RET
```

THE 'CLS COMMAND' ROUTINE (RUTINA 'COMANDA CLS')

Prima dată se 'sterge' tot ecranul - toti 'pixelii' se resetează si octetii atribut sînt setati pentru egalitatea cu valoarea ATTR-P - apoi partea inferioară a ecranului este refoarmată.

```
0D6B CLS          CALL  0DAF,CL-ALL    Se 'sterge' tot ecranul.
0D6E CLS-LOWER    LD    HL,+5C3C       Se încarcă TV-FLAG.
                  RES   5,(HL)         Semnal 'nu sterge partea
                                       inferioară a ecranului după
                                       apăsare tastă'.
                  SET   0,(HL)         Semnal 'aprte inferioară'.
                  CALL  0D4D,TEMPS     Se folosesc valorile
                                       permanente, adică se copiază
                                       ATTR-T din BORDCR.
                  LD    B,(DF-SZ)      Partea inferioară a ecranului
                  CALL  0E44,CL-LINE   este acum 'curătată' cu
                                       această valoare.
```

Cu exceptia octetilor atribut ai liniilor '22' & '23', octetii atribut ai liniilor din partea inferioară a ecranului trebuie să fie făcuti egali cu ATTR-P.

```
                  LD    HL,+5AC0       Octetul atribut la începutul
                                       liniei '22'.
                  LD    A,(ATTR-P)     Se încarcă ATTR-P.
                  DEC   B              Contorul de linie.
                  JR    0D8E,CLS-3     Salt înainte în buclă.
0D87 CLS-1        LD    C,+20          +20 de carcactere pe linie.
0D89 CLS-2        DEC   HL             Intoarcere de-a lungul liniei
                  LD    (HL),A         care are setat octetul
                                       atribut..
                  DEC   C
                  JR    NZ,0D89,CLS-2
0D8E CLS-3        DJNZ  0D87,CLS-1     Se ciclează pînă la terminare
```

Se poate fixa mărimea părtii inferioare a ecranului.

```
                  LD    (DF-SZ),+02    Vor fi două linii ca mărime.
```

Acum rămîne ca celelalte taskuri de 'administrare' să lucreze.

```
0D94 CL-CHAN      LD    A,+FD          Se deschide canalul 'K'.
                  CALL  1601,CHAN-OPEN
                  LD    HL,(CURCHL)    Se aduce adresa canalului
                  LD    DE,+09F4       actual si adresa de iesire
                  AND   A              este +09F4 (=PRINT-OUT) iar
0DA0 CL-CHAN-A    LD    (HL),E         adresa de intrare este +10A8
                  INC   HL             (=KEY-INPUT).
                  LD    (HL),D
                  INC   HL
                  LD    DE,+10A8
```

```
                CCF                     Prima dată adresa de iesire,
                JR    C,0DA0,CL-CHAN-A  apoi adresa de intrare.
                LD    BC,+1721          Cum partea inferioară a
                                        ecranului a fost tratată,
                                        'linia inferioară de afisare'
                                        va fi linia '23'.
                JR    0DD9,CL-SET       Revenire prin CL-SET.
```

THE 'CLEARING THE WHOLE DISPLAY AREA' SUBROUTINE (SUBRUTINA 'STERGEREA INTREGULUI SPATIU DE AFISARE')
Această subrutină este apelată: i. de către rutina de comandă CLS, ii. de către execuția principală a rutinei, iii. de către rutina de listare automată.

```
0DAF CL-ALL     LD    HL,+0000          Sistemul COORDS este resetat
                LD    (COORDS),HL       pe zero.
                RES   0,(FLAGS2)        Semnal 'ecranul este sters'.
                CALL  0D94,CL-CHAN      Prelucrare taskuri
                                        'administrare'.
                LD    A,+FE             Se deschide canalul 'S'.
                CALL  1601,CHAN-OPEN
                CALL  0D4D,TEMPS        Utilizare valori 'permanente'
                LD    B,+18             Acum se 'sterg' 24 de linii
                CALL  0E44,CL-LINE      de ecran.
                LD    HL,(CURCHL)       Se asigură că adresa curentă
                LD    DE,+09F4          de iesire este +09F4
                LD    (HL),E            (PRINT-OUT).
                INC   HL
                LD    (HL),D
                LD    (SCR-CT),+01      Resetare contor rulare.
                LD    BC,+1821          Cum partea superioară a
                                        ecranului a fost tratată,
                                        'linia de imprimare
                                        superioară' va fi linia '0'.
                                        Se continuă în CL-SET.
```

THE 'CL-SET' SUBROUTINE (SUBUTINA 'CL-SET')
Această subrutină este introdusă prin registrul pereche BC,care contine numerele de linie si coloană pentru zona caracterului, sau prin registrul C, care contine numărul coloanei din bufferul printer-ului. Adresa potrivită primului bit de caracter este găsită. Subrutina revine prin PO-STORE pentru a stoca toate valorile în sistemul de variabile cerut.

```
0DD9 CL-SET     LD    HL,+5B00          Inceputul buffer-ului
                                        printer-ului.
                BIT   A,(FLAGS)         Salt înainte pentru tratare
                JR    NZ,0DF4,CL-SET-2  buffer printer.
                LD    A,B               Se transferă numărul liniei.
                BIT   0,(IY-FLAG)       Salt înainte dacă se tratează
                JR    Z,0DEE,CL-SET-1   partea principală a ecranului
                ADD   A,(DF-SZ)         Linia de sus a părtii
                SUB   +18               superioare a ecranului este
                                        numită 'linia +18' si ea
                                        trebuie convertită.
0DEE CL-SET-1   PUSH  BC                Se salvează numerele liniei
                                        si coloanei.
                LD    B,A               Se încarcă numărul liniei.
                CALL  0E9B,CL-ADDR      Adresa de început a liniei
                                        este formată în HL.
                POP   BC                Numerele liniei si coloanei
                                        sînt aduse înapoi.
0DF4 CL-SET-2   LD    A,+21             Numărul coloanei este acum
                SUB   C                 inversat si transferat în
                LD    E,A               registrul pereche DE.
                LD    D,+00
                ADD   HL,DE             Adresa cerută este formată;
                JP    0A8C,PO-STORE     si adresa si numerele liniei
                                        si coloanei sînt stocate prin
                                        salt la PO-STORE.
```

THE 'SCROLLING' SUBROUTINE (SUBRUTINA 'RULARE'/'DEFILARE')
Numărul de linii ale ecranului ce trebuie rulate trebuie aduse, la intrarea în subrutina principală, în registrul B.

```
0DFE CL-SC-ALL  LD    B,+17             Punctul de intrare după
                                        'rulare'?
```

Punctul de intrare principal - de mai sus si fiind rularea se face pentru INPUT..AT.

```
0E00 CL-SCROLL    CALL  0E9B,CL-ADDR      Găseste adresa de început a
                                          liniei.
                  LD    C,+08             Sînt opt linii pixel pentru
                                          o linie completă.
```

Acum se intră în bucla principală de rulare. Registrul B contine numărul liniei superioare ce trebuie rulată, registrul pereche HL contine adresa de început a acestei linii din spatiul ecranului si registrul C contine pixelul de linie pentru culoare.

```
0E05 CL-SCR-1     PUSH  BC                Se salvează ambele contoare.
                  PUSH  HL                Se salvează adresa de
                                          început.
                  LD    A,B               Salt înainte numai dacă se
                  AND   +07               lucrează cu o 'treime' din
                  LD    A,B               ecran.
                  JR    NZ,0E19,CL-SCR-3
```

Pixelul de linie pentru linia de sus a 'treimii' ecranului trebuie mutată peste 2K limite. (Fiecare treime 2K.)

```
0E0B CL-SCR-2     EX    DE,HL             Rezultatul acestei mutări
                  LD    HL,+F8E0          lasă HL neschimbat iar DE va
                  ADD   HL,DE             indica destinatia cerută.
                  EX    DE,HL
                  LD    BC,+0020          Sînt +20 de caractere.
                  DEC   A                 Se decrementează contorul
                                          astfel încît o linie să fie
                                          tratată ca atare.
                  LDIR                    Acum se mută cei 32 de octeti
```

In continuare se prezintă pixelul liniilor prin care 'treimile' pot fi rulate. Registrul A contine, la primul pas, +01 - +07, +09 - +0F sau +11 - +17.

```
0E19 CL-SCR-3     EX    DE,HL             Se face ca DE să indice
                  LD    HL,+FFE0          destinatia cerută.
                  ADD   HL,DE             De această dată doar 32 de
                  EX    DE,HL             locatii avansează.
                  LD    B,A               Se salvează numărul liniei în
                                          registrul B.
                  AND   +07               Acum se determină cîte
                  RRCA                    caractere rămîn în 'treime'.
                  RRCA
                  RRCA
                  LD    C,A               Se aduce 'caracterul total'
                                          în registrul C.
                  LD    A,B               Se încarcă numărul liniei.
                  LD    B,+00             BC contine 'caracterul total'
                  LDIR                    si un pixel linie din fiecare
                                          din caracterele 'rulate'.
                  LD    B,+07             Acum se pregăteste pentru
                                          incrementare adresa de salt
                                          la a linita 'treimii'.
                  ADD   HL,BC             Se incrementează HL cu +0700.
                  AND   +F8               Salt înapoi dacă mai există o
                  JR    NZ,0E0D,CL-SCR-2  'treime de luat în
                                          considerare.
```

Acum se determină dacă bucla a fost parcursă de opt ori - o dată pentru fiecare pixel de linie.

```
                  POP   HL                Se încarcă adresa originară.
                  INC   H                 Se adresează următorul pixel
                                          linie.
                  POP   BC                Se aduc contorii.
                  DEC   C                 Decrementare contor pixel
                  JR    NZ,0E05,CL-SR-1   linie si salt înapoi numai
                                          dacă s-au mutat opt linii.
```

Urmează rularea octetilor atribut . De notat că registrul B contine încă numărul liniilor ce trebuie rulate si registrul C contine zero.

```
                  CALL  0E88,CL-ATTR      Sînt găsite adresele cerute
                                          în spatiul distribuit si
                                          numerele caracterelor în
                                          linia 'B'.
                  LD    HL,+FFE0          Deplasamentul pentru toti
                  ADD   HL,DE             octetii atribut este avansat
                  EX    DE,HL             cu 32 de locatii.
```

```
                LDIR                    Octetii atribut sînt
                                        'rulati'.
```

A rămas acum de curàtat doar linia de jos ale ecranului.

```
                LD    B,+01             Registrul B este încărcat cu
                                        +01 si CL-LINE este
                                        introdusă.
```

THE 'CLEAR LINES' SUBROUTINE (SUBRUTINA 'STERGERE LINII')
Această subrutină va sterge 'B' linii de jos ale ecranului.

```
0E44 CL-LINE    PUSH  BC                Se salvează numărul liniei pe
                                        durata acestei subrutine.
                CALL  0E9B,CL-ADDR      Adresa de început a liniei
                                        este formată în HL.
                LD    C,+08             Din nou trebuie considerate
                                        opt linii pixel.
```

Acum urmează o buclă de stergere a tuturor pixelilor linie.

```
0E4A CL-LINE-1  PUSH  BC                Salvare număr liniei si
                                        contor pixel linie.
                PUSH  HL                Salvare adresă.
                LD    A,B               Salvare număr linie în A.
0E4D CL-LINE-2  AND   +07               Se determină acum cîte
                RRCA                    cîte caractere sînt implicate
                                        în 'B mod 8' linii.
                RRCA                    Se trece rezultatul în
                RRCA                    registrul C. (C va conține
                LD    C,A               +00, acesta este 256 zec.,
                                        pentru o 'treime'.)
                LD    A,B               Se încarcă numărul liniei.
                LD    B,+00             Se face ca registrul pereche
                DEC   C                 BC să conțină 'cu unul mai
                                        puțin' decît numărul
                                        caracterelor.
                LD    D,H               Se face ca DE să indice
                LD    E,L               primul caracter.
                LD    (HL),+00          Se sterge octetul-pixel al
                                        primului caracter.
                INC   DE                Se face ca DE să indice al
                LDIR                    doilea caracter si se sterg
                                        octetii-pixel ale tuturor
                                        celorlalte caractere.
                LD    DE,+0701          Pentru fiecare 'treime' a
                ADD   HL,DE             ecranului HL se va incrementa
                                        cu +0701.
                DEC   A                 Acum se decrementează numărul
                                        liniei.
                AND   +F8               Se înlătură orice linie
                LD    B,A               superioară si se trece
                                        contorul 'treimii' în B.
                JR    NZ,0E4D,CL-LINE-2 Salt înapoi dacă mai există
                                        treimi cu care să se lucreze.
```

Acum se determină dacă bucla a fost folosită de opt ori.

```
                POP   HL                Se ridică adresa pentru
                INC   H                 fiecare pixel linie.
                POP   BC                Încărcare contori.
                DEC   C                 Decrementare contor pixel
                JR    NZ,0E4A,CL-LINE-1 linie si salt înapoi pînă se
                                        termină.
```

In continuare se setează octetii atribut, asa cum se cere. Valoarea din ATTR-P va fi folosită cînd se tratează partea principală a ecranului iar valoarea din BORDCR va fi folosită cînd se tratează partea inferioară.

```
                CALL  0E88,CL-ATTR      Se determină adresa primului
                                        octet atribut si numărul
                                        octetilor.
                LD    H,D               HL va indica primul octet
                LD    L,E               atribut iar DE îl va indica
                INC   DE                pe cel de al doilea.
                LD    A,(ATTR-P)        Se aduce valoarea din ATTR-P.
                BIT   0,(TV-FLAG)       Salt înainte dacă se tratează
                JR    Z,0E80,CL-LINE-3  partea principală a
                                        ecranului.
```

```
                    LD    A,(BORDCR)        Altfel, în loc se foloseste
                                            BORDCR.
0E80 CL-LINE-3      LD    (HL),A            Setare octet atribut.
                    DEC   BC                S-a făcut un octet.
                    LDIR                    Acum se copiază valoarea
                                            în toti octetii atribut.
                    POP   BC                Se reface numărul liniei.
                    LD    C,+21             Setare număr coloană la
                    RET                     stînga si revenire.
                    RET
```

THE 'CL-ATTR' SUBROUTINE (SUBRUTINA 'CL-ATTR)
Această subrutină îndeplineste două functii diferite.

i. Pentru o adresă dată a zonei ecranului, adresa proprie a atributului este returnată în registrul pereche DE. Este de notat faptul că valoarea indicatorului de intrare este pe a 'noua' liniei a a caracterului.

ii. Pentru un număr de linie dat, în registrul B, numărul zonelor caracterului de pe ecran de la începutul acestei linii în jos este readusă în registrul pereche BC.

```
0E88 CL-ATTR        LD    A,H               Se aduce bitul superior.
                    RRCA                    Se multiplică această valoare
                    RRCA                    de 32 de ori.
                    RRCA
                    DEC   A                 Intoarcere la linia a 'opta'.
                    OR    +50               Adresa zonei atribut.
                    LD    H,A               Se readuce octetul superior
                    EX    DE,HL             si se transferă adresa în DE.
                    LD    H,C               Este întotdeauna zero.
                    LD    L,B               Numărul liniei.
                    ADD   HL,HL             Se multplică de 32 de ori.
                    ADD   HL,HL
                    ADD   HL,HL
                    ADD   HL,HL
                    ADD   HL,HL
                    LD    B,H               Se mută rezultatul în
                    LD    C,L               registrul pereche BC înainte
                    RET                     de revenire.
```

THE 'CL-ADDR' SUBROUTINE (SUBRUTINA 'CL-ADDR')
Pentru un număr de linie dat, în registrul B, adresa cîmpului caracteristic ecranului este format în registrul pereche HL.

```
0E9B CL-ADDR        LD    A,+18             Numărul liniei trebuie
                    SUB   B                 inversat.
                    LD    D,A               Rezultatul este salvat în D.
                    RRCA                    Ca efect '(A mod 8) * 32'.
                    RRCA                    In o 'treime' a ecranului
                    RRCA                    octetul inferior pentru:
                    AND   +E0               linia 1 = +00
                                            linia 2 = +20, etc.
                    LD    L,A               Octetul inferior se transferă
                                            în L.
                    LD    A,D               Se încarcă numărul real al
                                            liniei.
                    AND   +18               Ca efect '64 + 8 * INT (A/8)'
                    OR    +40               Pentru 'treimea' de sus a
                                            ecranului octetul superior =
                                            +40, 'treimea' mijlocie =+48,
                                            iar pentru 'treimea' de jos
                                            = +50.
                    LD    H,A               Octetul superior este
                                            transferat în H.
                    RET                     Terminare.
```

THE 'COPY' COMMAND ROUTINE (RUTINA DE COMANDA 'COPY')
176 de pixeli linie ai ecranului sînt prelucrati rînd pe rînd.

```
0EAC COPY           DI                      Intreruperea mascabilă este
                                            întreruptă atîta timp cît
                                            durează COPY.
                    LD    B,+B0             '176' linii.
                    LD    HL,+4000          Adresa de bază a ecranului.
```

Acum se introduce următoarea buclă.

```
0EB2 COPY-1     PUSH  HL                  Se salvează adresa de bază si
                PUSH  BC                  numărul liniei.
                CALL  0EF4,COPY-LINE      Se apelează de '176' ori.
                POP   BC                  Se aduce numărul liniei si
                POP   HL                  adresa de bază.
                INC   H                   Se măreste adresa de bază cu
                                          '256' locatii pentru fiecare
                                          linie de pixeli.
                LD    A,H                 Salt înainte si rotire prin
                AND   +07                 buclă pentru opt linii pixel
                JR    NZ,0EC9,COPY-2      pentru o linie caracter.
```

Pentru fiecare linie de caracter nouă adresa de bază trebuie mărită.

```
                LD    A,L                 Se aduce octetul inferior.
                ADD   A,+20               Se măreste cu +20 octeti.
                LD    L,A                 Fanionul de transport va fi
                                          resetat cînd se lucrează 'în
                                          cadrul treimii'.
                CCF                       Se schimbă fanionul de
                                          transport.
                SBC   A,A                 Registrul A va contine +F8
                AND   +F8                 cînd se lucrează în cadrul
                                          'treimii' si +00 cînd o nouă
                                          'treime' este atinsă'.
                ADD   A,H                 Se măreste acum octetul
                LD    H,A                 superior al adresei.
0EC9 COPY-2     DJNZ  0EB2,COPY-1         Salt înapoi înă cînd vor fi
                                          afisate '176' de linii.
                JR    0EDA,COPY-END       Salt înainte la sfîrsitul
                                          rutinei.
```

THE 'COPY-BUFF' SUBROUTINE (SUBRUTINA 'COPY-BUFF')
Această subrutină este apelată întotdeauna cînd bufferul de imprimare trebuie să-si transfere continutul la afisaj.

```
0ECD COPY-BUFF  DI                        Dezactivare întrerupere
                                          mascabilă.
                LD    HL,+5B00            Adresa de bază a bufferului
                                          de imprimare.
                LD    B,+08               Sînt opt linii pixel.
0ED3 COPY-3     PUSH  BC                  Se salvează numărul liniei.
                CALL  0EF4,COPY-LINE      Se cheamă de '8' ori.
                POP   BC                  Se aduce numărul liniei.
                DJNZ  0ED3,COPY-3         Salt înapoi pînă cînd '8'
                                          linii au fost imprimate.
```

Continuare la rutina COPY-END.

```
0EDA COPY-END   LD    A,+04               Oprire imprimantă.
                OUT   (+FB),A
                EI                        Se activează întreruperea
                                          mascabilă si se continuă la
                                          CLEAR-PRB.
```

THE 'CLEAR PRINTER BUFFER' SUBROUTINE (SUBRUTINA 'STERGERE BUFFER IMPRIMARE')
Bufferul de imprimare este sters prin apelarea acestei subrutine.

```
0EDF CLEAR-PRB  LD    HL,+5B00            Adresa de bază a bufferului
                                          de imprimare.
                LD    (PR-CC),L           Resetare 'coloană' imprimator
                XOR   A                   Se sterge registrul A.
                LD    B,A                 De asemenea se sterge
                                          registrul B (de fapt B
                                          contine 256 zecimal).
0EE7 PRB-BYTES  LD    (HL),A              Toti cei '256' de octeti ai
                INC   HL                  bufferului de imprimare sînt
                DJNZ  0EE7,PRB-BYTES      stersi pe rînd.
                RES   1,(FLAGS2)          Semnal 'bufferul este gol'.
                LD    C,+21               Se setează poziția
                JP    0DD9,CL-SET         imprimatorului si se revine
                                          prin CL-SET & PO-STORE.
```

THE 'COPY-LINE' SUBROUTINE (SUBRUTINA 'COPY-LINE')
La intrarea în subrutină registrul pereche HL contine adresa de bază pentru cei 32 octeti care formează linia de pixeli si registrul B contine numărul pixelilor linie.

```
0EF4 COPY-LINE    LD    A,B          Se copiază numărul pixel-linie.
                  CP    +03          Registrul A va contine +00
                  SBC   A,A          pînă cînd ultimele două linii
                  AND   +02          vor fi tratate.
                  OUT   (+FB),A      Pornire motor doar pentru
                                     ultimele două linii
                                     superioare.
                  LD    D,A          Registrul A va contine sau
                                     +00 sau +02.
```

Mai trebuie făcute trei teste înaintea oricărei 'imprimări'.

```
0EFD COPY-L-1     CALL  1F54,BREAK-KEY     Salt înainte doar dacă nu s-a
                  JR    C,0F0C,COPY-L-2    apăsat tasta BREAK.
                  LD    A,+04              Dar dacă s-a apăsat tasta,
                  OUT   (+FB),A            se opreste motorul,
                  EI                       se activează întreruperile
                                           mascabile,
                  CALL  0EDF,CLEAR-PRB     se sterge bufferul de
                                           imprimare si
                  RST   0008,ERROR-1       se iasă prin rutina de
                  DEFB  +0C                tratare a erorilor - 'BREAK-
                                           CONT se repetă'.
0F0C COPY-L-2     IN    A,(+FB)            Se aduce starea imprimantei.
                  ADD   A,A
                  RET   M                  Dacă nu este prezentă
                                           imprimanta, se face imediat
                                           o revenire.
                  JR    NC,0EFD,COPY-L-1   Asteptare mod.
                  LD    C,+20              Sînt 32 de octeti.
```

Acum urmează o bucla care tratează acesti octeti.

```
0F14 COPY-L-3     LD    E,(HL)       Se aduce un octet.
                  INC   HL           Incrementare indicator.
                  LD    B,+08        Opt biti într-un octet.
0F18 COPY-L-4     RL    D            Se mută continutul lui D la
                                     stînga.
                  RL    E            Se mută fiecare bit in carry.
                  RR    D            Se mută continutul lui D
                                     înapoi, aducînd bitul de
                                     transport din E.
0F1E COPY-L-5     IN    A,(+FB)      Se aduce din nou starea
                  RRA                imprimantei si se asteaptă
                  JR    NC,0F1E,COPY-L-5  semnal de la decodor.
                  LD    A,B          Acum se merge mai departe si
                  OUT   (+FB),A      se trece 'bitul' la
                                     imprimantă.
                                     Notă: bit 2 - pornirea
                                     imprimantei, bit 1 -
                                     cresterea cu viteză mică a
                                     imprimării si bitul 7 este
                                     pentru ridicarea vitezei la
                                     actuala imprimare.
                  DJNZ  0F18,COPY-L-4      Se 'imprimă' fiecare bit.
                  DEC   C            Se decrementează contorul de
                                     octeti.
                  JR    NZ,0F14,COPY-L-3   Salt înapoi atîta vreme cît
                  RET                mai există octeti; altfel
                                     revenire.
```

THE 'EDITOR' ROUTINES (RUTINA 'EDITOR')

Editorul este apelat în două cazuri:

i. Din rutina de principală de executie astfel încît utilizatorul poate introduce o linie BASIC în sistem.

ii. Din rutina de comandă INPUT.

Prima dată se salvează 'eroare indicator stivă' iar apoi se prevede o adresă alternativă.

```
0F2C EDITOR       LD    HL,(ERR-SP)  Valoarea curentă este salvată
                  PUSH  HL           în stiva calculatorului.
0F30 ED-AGAIN     LD    HL,+107F     Acesta este ED-ERROR.
                  PUSH  HL           Orice eveniment care conduce
                  LD    (ERR-SP),SP  la rutina de tratare a
                                     erorii ce a fost folosită, va
                                     reveni la ED-ERROR.
```

Acum se introduce o buclă ce va trata apăsarea fiecărei taste.

```
0F38 ED-LOOP    CALL  15D4,WAIT-KEY      Revenire dacă s-a apăsat o
                                         tastă.
                PUSH  AF                 Se salvează temporar codul.
                LD    D,+00              Se aduce durata clicului
                LD    E,(PIP)            tastei.
                LD    HL,+00C8           Si a căderii.
                CALL  03B5,BEEPER        Acum se realizează 'beep'-ul.
                POP   AF                 Refacerea codului.
                LD    HL,+0F38           Adresa ED-LOOP este încărcată
                PUSH  HL                 în stiva calculatorului.
```

Acum se face analiza codului obtinut.

```
                CP    +18                Se acceptă toate codurile
                JR    NC,0F81,ADD-CHAR   caracterelor, caracterelor
                                         grafice si ale semnelor.
                CP    +07                De asemenea se acceptă ','.
                JR    C,0F81,ADD-CHAR
                CP    +10                Salt înainte dacă codul
                JR    C,0F92,ED-KEYS     reprezintă o tastă de editare
```

Acum se iau în considerare tastele de control -- de la INK la TAB.

```
                LD    BC,+0002           INK & PAPER necesită două
                                         locatii.
                LD    D,A                Se copiază codul în D.
                CP    +16                Salt înainte cu 'INK&PAPER'.
                JR    C,0F6C,ED-CONTR
```

AT & TAB se tratează după cum urmează:

```
                INC   BC                 Se cer trei locatii.
                BIT   7,(FLAGX)          Salt înainte dacă nu se
                JP    Z,101E,ED-IGNORE   lucrează cu INPUT LINE...
                CALL  15D4,WAIT-KEY      Se aduce cel de al doilea cod
                LD    E,A                si se încarcă în E.
```

Acum se încarcă ceilalti octeti pentru controlul caracterelor.

```
0F6C ED-CONTR   CALL  15D4,WAIT-KEY      Se aduce alt cod.
                PUSH  DE                 Se salvează codul anterior.
                LD    HL,(K-CUR)         Se aduce K-CUR.
                RES   0,(MODE)           Semnal 'mod K'.
                CALL  1655,MAKE-ROOM     Se realizează două sau trei
                                         spatii.
                POP   BC                 Se readuce codul anterior.
                INC   HL                 Se indică prima locatie.
                LD    (HL),B             Se introduce primul cod.
                INC   HL                 Apoi se introduce al doilea
                LD    (HL),C             cod care va fi scris
                                         deasupra dacă sînt numai două
                                         coduri - aceasta cu INK&PAPER
                JP    0F8B,ADD-CH-1      Salt înainte.
```

THE 'ADD-CHAR' SUBROUTINE (SUBRUTINA 'ADD-CHAR')
Această subrutină de obicei adună un cod la linia curentă EDIT (editare) sau INPUT (introducere).

```
0F81 ADD-CHAR   RES   0,(MODE)           Semnal 'mod K'.
                LD    HL,(K-CUR)         Se aduce pozitia cursorului.
                CALL  1652,ONE-SPACE     Se realizează un singur
                                         spatiu.
0F8B ADD-CH-1   LD    (DE),A             Se introduce codul în spatiu
                INC   DE                 si se semnalează că cursorul
                LD    (K-CUR),DE         se pozitionează cu o locatie
                RET                      după. Apoi se revine indirect
                                         în ED-LOOP.
```

Tastele de comandă sînt tratate ca mai jos:

```
0F92 ED-KEYS    LD    E,A                Codul este transferat în
                LD    D,+00              registrul pereche DE.
                LD    HL,+0F99           Adresa de bază a tabelului
                                         tastei de editare.
                ADD   HL,DE              Intrarea este adresată si
                LD    E,(HL)             apoi este introdusă în DE.
```

```
ADD   HL,DE          Adresa rutinei de tratare
PUSH  HL             este salvatá în stiva
                     calculatorului.
LD    HL,(K-CUR)     Registrul pereche HL este
RET                  setat si se face un salt
                     indirect la rutina cerutá.
```

THE 'EDITING KEYS' TABLE (TABELUL 'TASTELOR DE EDITARE')

adresá	deplasament	caracter	adresá	deplasament	caracter
0FA0	09	EDIT	0FA5	70	DELETE
0FA1	66	cursor stînga	0FA6	7E	ENTER
0FA2	6A	cursor dreapta	0FA7	CF	SYMBOL SHIFT
0FA3	50	cursor jos	0FA8	D4	GRAPHICS
0FA4	B5	cursor sus			

THE 'EDIT KEY' SUBROUTINE (SUBRUTINA 'EDITARE TASTA')

Cînd în 'mod editare' se apasá tasta EDIT se va aduce jod 'linia BASIC actualá'.
In 'mod INPUT' actiunea tastei EDIT este de a sterge rásunsul curent si de a încuviinta unul nou.

```
0FA9 ED-EDIT    LD    HL,(E-PPC)       Se aduce numárul liniei
                                       curente.
                BIT   5,(FLAGX)        Salt înainte dacá este
                JP    NZ,1097,CLEAR-SP în 'mod INPUT'.
                CALL  196E,LINE-ADDR   Gáseste adresa de început a
                CALL  1695,LINE-NO     liniei curente si avanseazá
                                       numárul ei.
                LD    A,D              Dacá numárul liniei returnate
                OR    E                este zero, atunci pur si
                JP    Z,1097,CLEAR-SP  simplu se sterge zona de
                                       editare.
                PUSH  HL               Se salveazá adresa liniei.
                INC   HL               Se pástreazá lungimea liniei.
                LD    C,(HL)
                INC   HL
                LD    B,(HL)
                LD    HL,+000A         Se adunà +0A la lungime si se
                ADD   HL,BC            testeazá pentru a determina
                LD    B,H              dacá sînt suficiente locatii
                LD    C,L              pentru o copie a liniei.
                CALL  1F05,TEST-ROOM
                CALL  1097,CLEAR-SP    Acum se sterge spatiul de
                                       editare.
                LD    HL,(CURCHL)      Se aduce adresa canalului
                EX    (SP),HL          curent si se schimbá cu
                                       adresa liniei.
                PUSH  HL               Salvare temporará a acesteia.
                LD    A,+FE            Se deschide canalul 'R' asa
                CALL  1601,CHAN-OPEN   cá linia va fi copiatá în
                                       zona de editare.
                POP   HL               Se aduce adresa liniei.
                DEC   HL               Se trece înainte de linie.
                DEC   (E-PPC-lo)       Se decrementeazá numárul
                                       liniei curente astfel încît
                                       sá se evite tiparirea peste
                                       cursor.
                CALL  1855,OUT-LINE    Se tipáreste linia BASIC.
                INC   (E-PPC-lo)       Se incrementeazá numárul
                                       liniei curente.
                                       Notá: Decrementarea numárului
                                       liniei nu evitá întotdeauna
                                       suprainprimarea cursorului.
                LD    HL,(E-LINE)      Se aduce începutul liniei în
                INC   HL               spatiul de editare si se
                INC   HL               trece dincolo de numárul
                INC   HL               liniei si de lungime pentru a
                INC   HL               gási adresa lui K-CUR.
                LD    (K-CUR),HL
                POP   HL               Se aduce adresa canalului
                CALL  1615,CHAN-FLAG   anterior si se seteazá
                RET                    fanioanele corespunzátoare
                                       înainte de a reveni în
                                       ED-LOOP.
```

THE 'CURSOR DOWN EDITING' SUBROUTINE (SUBRUTINA 'EDITARE CURSOR JOS')

```
0FF3 ED-DOWN    BIT   5,(FLAGX)          Salt înainte dacă este 'mod
                JR    NZ,1001,ED-STOP    INPUT'.
                LD    HL,+5C49           Aceasta este E-PPC.
                CALL  190F,LN-FETCH      S-a găsit numărul liniei
                JR    106E,ED-LIST       următoare și se realizează o
                                         nouă listare automată.
1001 ED-STOP    LD    (ERR-NR),+10       Prezentarea 'STOP în INPUT'.
                JR    1024,ED-ENTER      Salt înainte.

THE 'CURSOR LEFT EDITING' SUBROUTINE (SUBRUTINA 'EDITARE CURSOR STINGA')

1007 ED-LEFT    CALL  1031,ED-EDGE       Cursorul este mutat.
                JR    1011,ED-CUR        Salt înainte.
```

THE ' CURSOR RIGHT EDITING' SUBROUTINE (SUBRUTINA 'EDITARE CURSOR DREAPTA')

```
100C ED-RIGHT   LD    A,(HL)             Caracterul curent este testat
                CP    +0D                si se execută revenire dacă
                RET   Z                  este 'accriage return'.
                INC   HL                 In caz contrar se determină
                                         poziționarea cursorului după
                                         caracter.
1011 ED-CUR     LD    (K-CUR),HL         Se setează variabilă sistem
                                         K-CUR.
                RET
```

THE 'DELETE EDITING' SUBROUTINE (SUBRUTINA 'EDITARE STERGERE')

```
1015 ED-DELETE  CALL  1031,ED-EDGE       Se mută cursorul la stînga.
                LD    BC,+0001           Cerere caracter curent.
                JP    19E8,RECLAIM-2
```

THE 'ED-IGNORE' SUBROUTINE (SUBRUTINA 'ED-IGNORE' (ignorare editare))

```
101E ED-IGNORE  CALL  15D4,WAIT-KEY      Următoarele două coduri ale
                CALL  15D4,WAIT-KEY      rutinei de introducere tastă
                                         se ignoră.
```

THE 'ENTER EDITING' SUBROUTINE (SUBRUTINA 'INTRODUCERE EDITARE')

```
1024 ED-ENTER   POP   HL                 Se înlătură adresele de la
                POP   HL                 ED-LOOP și ED-ERROR.
1026 ED-END     POP   HL                 Se reface vechea valoare a
                LD    (ERR-SP),HL        lui ERR-SP.
                BIT   7,(ERR-NR)         Revenire dacă nu au fost
                RET   NZ                 erori.
                LD    SP,HL              Altfel se execută un salt
                RET                      indirect la rutina de eroare.
```

THE 'ED-EDGE' SUBROUTINE (SUBRUTINA 'ED-EDGE')
Adresa cursorului se află în registrul pereche HL și va fi decrementată dacă cursorul nu se află la începutul liniei. Trebuie avut grijă ca să nu se pună cursorul între caracterele de control și parametrii lor.

```
1031 ED-EDGE    SCF                      DE va conține fie E-LINE
                CALL  1195,SET-DE        (pentru editare) fie WORKSP
                                         (pentru introducere).
                SBC   HL,DE              Fanionul de transport va fi
                                         setat dacă cursorul este deja
                ADD   HL,DE              la începutul liniei.
                INC   HL                 Corectare pentru scădere.
                POP   BC                 Se trimite adresa de
                                         revenire.
                RET   C                  Revenire prin ED-LOOP dacă
                                         fanionul de transport este
                                         setat.
                PUSH  BC                 Se reface adresa de revenire.
                LD    B,H                Se mută adresa curentă a
                LD    C,L                cursorului în BC.
```

Acum se introduce o buclă care să verifice dacă caracterele de control nu sînt despărțite de parametrii lor.

```
103E ED-EDGE-1  LD    H,D                HL va indica caracterul în
                LD    L,E                linie după cea adresată de DE
                INC   HL
```

```
                LD    A,(DE)          Se aduce codul caracterului.
                AND   +F0             Salt înainte dacă caracterul
                CP    +10             nu reprezintă INK to TAB.
                JR    NZ,1051,ED-EDGE-2
                INC   HL              Este permis un parametru.
                LD    A,(DE)          Se aduce iarăsi codul.
                SUB   +17             Carry este resetat pentru TAB
                ADC   A,+00           Notă: Aceasta desparte AT&TAB
                                      dar AT&TAB în această formă
                                      oricum nu sînt implementate,
                                      asa că nu implică nici o
                                      diferentiere.
                JR    NZ,1051,ED-EDGE-2   Salt înainte dacă nu se
                INC   HL              lucrează cu AT&TAB care vor
                                      avea doi parametrii, cînd se
                                      folosesc.
1051 ED-EDGE-2  AND   A               Pregătire pentru scădere.
                SBC   HL,BC           Fanionul de transport va fi
                ADD   HL,BC           resetat cînd 'indicatorul
                                      actualizat' ajunge pe K-CUR.
                EX    DE,HL           Pentru următoarea buclă se
                JR    C,103E,ED-EDGE-1    foloseste 'indicatorul
                RET                   actualizat', dar dacă există
                                      se foloseste 'indicatorul
                                      prezent' pentru K-CUR.
                                      Notă: Se sterge caracterul de
                                      control cînd este folosit
                                      DELETE.
```

THE 'CURSOR UP EDITING' SUBROUTINE (SUBRUTINA 'EDITARE CURSOR SUS')

```
1059 ED-UP      BIT   5,(FLAGX)       Revenire dacă este în 'mod
                RET   NZ              INPUT'.
                LD    HL,(E-PPC)      Se aduce numărul liniei
                CALL  196E,LINE-ADDR  curente si adresa sa de
                                      început.
                EX    DE,HL           HL indică acum linia
                                      anterioară.
                CALL  1695,LINE-NO    Se aduce acest număr de
                                      linie.
                LD    HL,+5C4A        Este E-PPC-hi.
                CALL  191C,LN-STORE   Se stochează numărul liniei.
106E ED-LIST    CALL  1795,AUTO-LIST  Se execută o nouă listare
                LD    A,+00           automată si canalul 'K' este
                JP    1601,CHAN-OPEN  redeschis înainte de a se
                                      reveni la ED-LOOP.
```

THE 'ED-SYMBOL' SUBROUTINE (SUBRUTINA 'ED-SYMBOL (editare simbol))
Dacă se vor folosi codurile SYMBOL & GRAPHICS, ele vor fi tratate cum sînt prezentate mai jos:

```
1076 ED-SYMBOL  BIT   7,(FLAGX)       Salt înapoi dacă nu se
                JR    Z,1024,ED-ENTER lucrează cu INPUT .. LINE.
1078 ED-GRAPH   JP    0F81,ADD-CHAR   Salt înapoi.
```

THE 'ED-ERROR' SUBROUTINE (SUBRUTINA 'ED-ERROR' (editare eroare))
Aici se va executa saltul în cazul în care au apărut diferite tipuri de erori.

```
107F ED-ERROR   BIT   4,(FLAGS2)      Salt înapoi dacă se foloseste
                JR    Z,1026,ED-END   alt canal decît canalul 'K'.
                LD    (ERR-NR),+FF    Se anulează numărul erorii si
                LD    D,+00           se dă o semnalizare sonoră
                LD    E,(BASP)        înainte de a intra iar în
                LD    HL,+1A90        editor.
                CALL  03B5,BEEPER
                JP    0F30,ED-AGAIN
```

THE ' CLEAR-SP' SUBROUTINE (SUBRUTINA 'CLEAR-SP' (stergere spatiu))
Zona de editare sau spatiul de lucru se sterge după cum se indică.

```
1097 CLEAR-SP   PUSH  HL              Salvare indicator de spatiu.
                CALL  1190,SET-HL     DE va indica primul caracter
                                      si HL pe ultimul.
                DEC   HL              Acum se cere valoarea
                CALL  19E5,RECLAIM-1  corectă.
                LD    (K-CUR),HL      Variabilele sistem K-CUR si
                LD    (MODE),+00      MODE ('mod K') sînt
                POP   HL              initializate înainte de a
                RET                   se aduce indicatorul, apoi
```

revenire.

THE 'KEYBOARD INPUT' SUBROUTINE (SUBRUTINA 'INTRODUCERE TASTATURA')
Această importantă subrutină readuce codul ultimei taste care a fost apăsată, si este de notat faptul că CAPS LOCK, schimbarea modului si parametrii de control ai culorii sînt prezentati în cadrul subrutinei.

```
10A8 KEY-INPUT   BIT  3,(IY-FLAG)        Copiere linie-editare sau
                 CALL NZ,111D,ED-COPY     linie-introducere pe ecran
                                          dacă modul a fost schimbat.
                 AND  A                   Revenire cu fanionul de
                 BIT  5,(FLAGS)           transport si de zero resetate
                 RET  Z                   dacă nici o tastă nu a fost
                                          apăsată.
                 LD   A,(LAST-K)          Altfel se încarcă codul si se
                 RES  5,(FLAGS)           semnalează că a fost luat.
                 PUSH AF                  Salvare temporară a codului.
                 BIT  5,(IY-FLAG)         Dacă este necesar se sterge
                 CALL NZ,0D6E,CLS-LOWER   partea inferioară a ecranului
                                          ex. după 'defilare?'.
                 POP  AF                  Se aduce codul.
                 CP   +20                 Se acceptă toate codurile de
                 JR   NC,111B,KEY-DONE    caractere si semne.
                 CP   +10                 Salt înainte cu majoritatea
                 JR   NC,10FA,KEY-CONTR   codurilor caracterelor de
                                          control.
                 CP   +06                 Salt înainte cu codurile
                 JR   NC,10DB,KEY-M&CL    'mod' si cu codul CAPS LOCK.
```

Acum se lucrează cu codurile FLASH, BRIGHT & INVERSE.

```
                 LD   B,A                 Se salvează codul.
                 AND  +01                 Se păstrează doar bitul 0.
                 LD   C,A                 C contine +00 (= OFF) sau
                                          C contine +01 (= ON).
                 LD   A,B                 Se aduce codul.
                 RRA                      Se roteste o dată (se pierde
                                          bitul 0).
                 ADD  A,+12               Se incrementează cu +12
                 JR   1105,KEY-DATA       rezultînd FLASH- +12, BRIGHT-
                                          +13 si INVERSE - +14.
```

Codul CAPS LOCK si codurile modului sînt lucrate cu 'limite'.

```
10DB KEY-M&CL    JR   NZ,10E6,KEY-MODE    Salt înainte cu codurile
                                          'modului'.
                 LD   HL,+5C6A            Aceasta este FLAGS2.
                 LD   A,+08               Se schimbă bitul 3 din FLAGS.
                 XOR  (HL)                Aceasta este fanionul pentru
                                          CAPS LOCK.
                 LD   (HL),A
                 JR   10F4,KEY-FLAG       Salt înainte.
10E6 KEY-MODE    CP   +0E                 Se verifică limita inferioară
                 RET  C
                 SUB  +0D                 Se reduce rangul.
                 LD   HL,+5C41            Incărcare MODE.
                 CP   (HL)                A fost schimbat?
                 LD   (HL),A              Se introduce cod 'mod' nou.
                 JR   NZ,10F4,KEY-FLAG    Salt dacă a fost schimbat;
                 LD   (HL),+00            altfel se schimbă în 'mod L'.
10F4 KEY-FLAG    SET  3,(IY-FLAG)         Semnal 'modul poate fi
                                          schimbat'.
                 CP   A                   Resetare fanion de transport
                 RET                      si revenire.
```

Se prelucrează codurile de control tastă (diferite de FLASH, BRIGHT & INVERSE)

```
10FA KEY-CONTR   LD   B,A                 Se salvează codul.
                 AND  +07                 Se face ca registrul C să
                 LD   C,A                 conțină parametrul (+00 la
                                          +07).
                 LD   A,+10               Registrul A contine acum
                                          codul INK.
                 BIT  3,B                 Dar dacă codul a fost unul
                 JR   NZ,1105,KEY-DATA    'nedeplasat', atunci se face
                 INC  A                   astfel ca A să conțină codul
                                          PAPER.
```

Parametrul este salvat în K-DATA si adresa canalului schimbată de la KEY-INPUT

la KEY-NEXT.

```
1105 KEY-DATA    LD   (K-DATA),C          Salvare parametru.
                 LD   DE,+110D            Se încarcă adresa KEY-NEXT.
                 JR   1113,KEY-CHAN       Salt înainte.
```

Notă: La primul pas la intrarea în KEY-INPUT, registrul A este returnat conținînd un 'cod de control' și apoi, la următorul pas, la intrarea în KEY-NEXT, acesta este parametrul care este returnat.

```
110D KEY-NEXT    LD   A,(K-DATA)          Se aduce parametrul.
                 LD   DE,+10A8            Se încarcă adresa KEY-INPUT.
```

Se setază adresa de intrare în prima zonă canal.

```
1113 KEY-CHAN    LD   HL,(CHANS)          Se aduce adresa canalului.
                 INC  HL
                 INC  HL
                 LD   (HL),E              Acum se setează adresa de
                                          intrare.
                 INC  HL
                 LD   (HL),D
```

In final se iese cu codul cerut în registrul A.

```
111B KEY-DONE    SCF                      Se arată că un cod a fost
                 RET                      găsit si se revine.
```

THE 'LOWER SCREEN COPYING' SUBROUTINE (SUBRUTINA 'COPIERE PARTEA INFERIOARA A ECRANULUI')
Această subrutină este apelată întotdeauna cînd linia din zona de editare sau din spatiul INPUT urmează să fie tipărită în partea de jos a ecranului.

```
111D ED-COPY     CALL 0D4D,TEMPS          Se utilizează culorile
                                          permanente.
                 RES  3,(TV-FLAG)         Se semnalizează că 'modul
                 RES  5,(TV-FLAG)         trebuie considerat
                                          neschimbat' si 'partea de
                                          jos a ecranului nu trebuie
                                          stearsă'.
                 LD   HL,(S-POSNL)        Se salvează valoarea curentă
                 PUSH HL                  a lui S-POSNL.
                 LD   HL,(ERR-SP)         Se păstrează valoarea curentă
                 PUSH HL                  a lui ERR-SP.
                 LD   HL,+1167            Adresa ED-FULL.
                 PUSH HL                  Se salvează această adresă în
                 LD   (ERR-SP),SPS        stiva calculatorului pentru a
                                          face ???
                 LD   HL,(ECHO-E)         Se încarcă stiva cu valoarea
                 PUSH HL                  lui ECHO-E.
                 SCF                      Se face ca HL să indice
                 CALL 1195,SET-HL         începutul spatiului iar DE
                 EX   DE,HL               sfîrsitul lui.
                 CALL 187D,OUT-LINE2      Acum se tipăreste linia.
                 EX   DE,HL               Se schimbă indicatorii si se
                 CALL 18E1,OUT-CURS       afisează cursorul.
                 LD   HL,(S-POSNL)        Apoi se aduce valoarea
                 EX   (SP),HL             curentă a lui S-POSNL si se
                                          schimbă cu ECHO-E.
                 EX   DE,HL               Se trece CEHO-E în DE.
                 CALL 0D4D,TEMPS          Se aduc din nou culorile
                                          permanente.
```

Restul oricărei linii care a fost începută este acum completată cu tipărirea spatiilor cu culoarea permanentă a hîrtiei (PAPER).

```
1150 ED-BLANK    LD   A,(S-POSNL-hi)      Se aduce numărul liniei
                 SUB  D                   curente si se scade numărul
                                          liniei vechi.
                 JR   C,117C,ED-C-DONE    Salt înainte dacă nu s-a
                                          cerut nici un 'spatiu' de
                                          linie.
                 JR   NZ,115E,ED-SPACES   Salt înainte dacă nu este pe
                                          aceeasi linie.
                 LD   A,E                 Se aduce numărul coloanei
                 SUB  (S-POSNL-lo)        vechi si se scade numărul
                                          coloanei noi.
                 JR   NC,117C,ED-C-DONE   Salt dacă nu este nici o
                                          cerere de spatiu.
```

```
115E ED-SPACES    LD    A,+20          Se încarcă un 'spatiu'.
                  PUSH  DE             Se salvează vechile valori.
                  CALL  09F4,PRINT-OUT Se tipăresc.
                  POP   DE             Se aduc vechile valori.
                  JR    1150,ED-BLANK  Din nou înapoi.
```

Acum se lucrează cu orice erori.

```
1167 ED-FULL      LD    D,+00          Semnalizare sonoră.
                  LD    E,(RASP)
                  LD    HL,+1A90
                  CALL  03B5,BEEPER
                  LD    (ERR-NR),+FF   Anulare număr eroare.
                  LD    BC,(S-POSNL)   Se aduce valoarea curentă a
                  JR    117E,ED-C-END  lui S-POSNL si se execută
                                       salt înainte.
117C ED-C-DONE    POP   DE             Valoarea noii pozitii.
                  POP   HL             'Adresa erorii'.
```

Aici se vine după o eroare.

```
117E ED-C-END     POP   HL             Se reface vechea valoare a
                  LD    (ERR-SP),HL    lui ERR-SP.
                  POP   BC             Se aduce vechea valoare a lui
                                       S-POSNL.
                  PUSH  DE             Se salvează valorile pozitiei
                                       noi.
                  CALL  0DD9,CL-SET    Se setează variabilele
                                       sistemului.
                  POP   HL             Vechea valoarea a lui S-POSNL
                  LD    (ECHO-E),HL    se încarcă în ECHO-E.
                  LD    (X-PTR-hi),+00 X-PTR se sterge în mai multe
                  RET                  feluri si se face revenire.
```

THE 'SET-HL' AND 'SET-DE' SUBROUTINES
Aceste subrutine revin cu HL indicînd prima locatie si DE 'ultima' locatie pentru fiecare spatiu de editare sau zonă de lucru.

```
1190 SET-HL       LD    HL,(WORKSP)    Este indicată ultima locatie
                  DEC   HL             a spatiului de editare.
                  AND   A              Se sterge fanionul de
                                       transport.
1195 SET-DE       LD    DE,(E-LINE)    Se indică începutul spatiului
                  BIT   5,(FLAGX)      de editare si se revine dacă
                  RET   Z              este 'mod editare'.
                  LD    DE,(WORKSP)    Altfel este schimbat DE.
                  RET   C              Se revine, dacă se
                                       intenționează.
                  LD    HL,(STKBOT)    Se aduce STKBOT si se revine.
                  RET
```

THE 'REMOVE-FP' SUBROUTINE
Această subrutină rearată punctul flotant ascuns format într-o linie BASIC.

```
11A7 REMOVE-FP    LD    A,(HL)         Se examinează pe rînd fiecare
                                       caracter.
                  CP    +0E            Este semnul unui număr?
                  LD    BC,+0006       Va ocupa sase locatii.
                  CALL  Z,19E8,RECLAIM-2 Este cerut numărul F-P.
                  LD    A,(HL)         Se aduce din nou codul.
                  INC   HL             Se măreste indicatorul.
                  CP    +0D            'Carriage return'?
                  JR    NZ,11A7,REMOVE-FP Înapoi dacă nu. Dar dacă
                  RET                  este se face o întoarcere
                                       simplă.
```

THE EXECUTIVE ROUTINES (RUTINELE EXECUTIVULUI)

THE 'INITIALISATION' ROUTINE (RUTINA 'INITIALIZARE')

Punctul principal de intrare în această rutină este la START/NEW (11CB). Cînde intrarea se face prin START (0000), ca si cînd atunci se pune sub tensiune sistemul, registrul A contine zero si registrul pereche DE contine valoarea +FFFF. Oricum, punctul principal de intrare poate fi atins si urmînd executia rutinei de comandă NEW.

THE 'NEW COMMAND' ROUTINE (RUTINA 'COMANDA NOUA')

```
11B7 NEW        DI                          Dezactivare întrerupere
                                            mascabilă.
                LD    A,+FF                 Fanionul NEW.
                LD    DE,(RAMTOP)           Se păstrează valoarea
                                            existentă a lui RAMTOP.
                EXX                         Se încarcă registrii
                LD    BC,(P-RAMT)           alternativi cu următoarele
                LD    DE,(RASP/PIP)         variabile sistem. Si acestea
                LD    HL,(UDG)              vor si păstrate.
```

Punctul principal de intrare.

```
11CB START/NEW  LD    BC                    Se salvează fanionul pentru
                                            mai tîrziu.
                LD    A,+07                 Se colorează marginea albă.
                OUT   (+FE),A
                LD    A,+3F                 Se încarcă registrul I cu
                LD    I,A                   valoarea +3F.
                DEFB  +00,+00,+00           Se asteaptă 24 de stări T.
                DEFB  +00,+00,+00
```

Acum se verifică memoria.

```
11DA RAM-CHEK   LD    H,D                   Se transferă valoarea în DE.
                LD    L,E                   (START = +FFFF, NEW = RAMTOP)
11DC RAM-FILL   LD    (HL),+02              Se introduce valoarea +02 în
                DEC   HL                    fiecare locatie pînă la
                CP    H                     +3FFF.
                JR    NZ,11DC,RAM-FILL
11E2 RAM-READ   AND   A                     Pregătire pentru scădere.
                SBC   HL,DE                 Fanionul de transport va fi
                ADD   HL,DE                 resetat cînd se atinge vîrful
                INC   HL                    Se măreste indicatorul.
                JR    NC,11EF,RAM-DONE      Salt cînd este în vîrf.
                DEC   (HL)                  +02 ajunge +01.
                JR    Z,11EF,RAM-DONE       Dacă ajunge zero, atunci RAM
                                            gresit.
                                            Se foloseste ca vîrf HL
                                            actual.
                DEC   (HL)                  +01 ajunge +00.
                JR    Z,11E2,RAM-READ       Se trece la următorul test
                                            dacă nu este eroare.
11EF RAM-DONE   DEC   HL                    HL indică ultima locatie
                                            curentă în ordinea lucrului.
```

Urmează refacerea sistemului de variabile 'păstrate'. (Ele sînt lipsite de sens dacă vin de la START.)

```
                EXX                         Se schimbă registrii.
                LD    (P-RAMT),BC           Se refac P-RAMT, RASP/PIP &
                LD    (RASP/PIP),DE         UDG.
                LD    (UDG),HL
                EXX
                INC   B                     Se testează fanionul START/
                                            NEW.
                JR    Z,1219,RAM-SET        Salt înainte dacă se vine de
                                            la rutina de comandă NEW.
```

Se redefinesc variabilele sistemului cînde se vine de la START si se initializează spatiul grafic definit de utilizator.

```
                LD    (P-RAMT),HL           Vîrf RAM fizic.
                LD    DE,+3EAF              Ultimul octet al lui 'U' în
                                            caracterul setat.
                LD    BC,+00A8              Există acest număr de octeti
                                            în 21 de litere.
                EX    DE,HL                 Schimbare indicatori.
                LDDR                        Acum se copiază forma
```

```
                                        caracterelor de litere de la
                                        'A' la 'U'.
                EX      DE,HL           Se schimbă indicatorii înapoi
                INC     HL              Se indică primul octet.
                LD      (UDG),HL        Acum se setează UDG.
                DEC     HL              Se scade o locatie.
                LD      BC,+0040        Se setează variabilele sistem
                LD      (RASP/PIP),BC   RASP & PIP.
```

Restul rutinei este comun si pentru operatiile START si pentru operatiile NEW.

```
1219 RAM-SET    LD      (RAMTOP),HL     Se setează RAMTOP.
                LD      HL,+3C00        Se intializează variabila
                LD      (CHARS),HL      sistem CHARS.
```

In continuare se setează stiva calulatorului.

```
                LD      HL,(RAMTOP)     Locatia din vîrf va contine
                LD      (HL),+3E        +3E.
                DEC     HL              Următoarea locatie este
                                        făcută să conțină zero.
                LD      SP,HL           Aceste două locatii
                                        reprezintă 'ultima intrare'.
                DEC     HL              Se coboară două locatii
                DEC     HL              pentru a afla valoarea
                LD      (ERR-SP),HL     corectă pentru ERR-SP.
```

Initializarea rutinei continuă cu:

```
                IM      1               Se foloseste modul 1 de
                                        întrerupere.
                LD      IY,+5C3A        IY contine întotdeauna
                                        +ERR-NR.
                EI                      Se poate activa acum
                                        întreruperea mascabilă.
                                        Ceasul de timp real va fi
                                        mărit si tastatura scanată
                                        la fiecare a 1/50 partea
                                        dintr-o secundă.
                LD      HL,+5CB6        Adresa de bază a zonei
                LD      (CHANS),HL      canalului de informatii.
                LD      DE,15AF         Canalul de informatii initial
                LD      BC,+0015        este mutat din tabel (15AF)
                EX      DE,HL           în zona canalului de
                LDIR                    informatii.
                EX      DE,HL           Variabila sistem DATADD este
                DEC     HL              făcută să arate ultima
                LD      (DATADD),HL     locatie a canalului de
                                        informatii.
                INC     HL              Iar PROG & VARS va indica
                LD      (PROG),HL       locatia următoare.
                LD      (VARS),HL
                LD      (HL),+80        Indicatorul de sfîrsit al
                                        spatiului variabilelor.
                INC     HL              Se incrementează cu o locatie
                LD      (E-LINE),HL     pentru a găsi valoarea pentru
                                        E-LINE.
                LF      (HL),+0D        Se face ca linia editată să
                INC     HL              conțină un singur 'carriage
                                        return'.
                LD      (HL),+80        Acum se introduce un marcator
                                        de sfîrsit.
                INC     HL              Se mută cu o locatie pentru
                LD      (WORKSP),HL     a se găsi valoarea pentru
                LD      (STKBOT),HL     WORKSP, STKBOT & STKEND.
                LD      (STKEND),HL
                LD      A,+38           Initializare variabile sistem
                LD      (ATTR-P),A      pentru culoare pe: FLASH 0,
                LD      (ATTR-T),A      BRIGHT 0, PAPER 7, & INK 0.
                LD      (BORDCR),A
                LD      HL,+0523        Se initializează variabilele
                LD      (REPDEL),HL     sistem REPDEL & REPPER.
                DEC     (KSTATE-0)      KSTATE-0 va contine +FF.
                DEC     (KSTATE-4)      KSTATE-4 va contine +FF.
                LD      HL,+15C6        In continuare se mută sirul
                LD      DE,+5C10        initial de informatii din
                LD      BC,+000E        tabelul său în spatiul
                LDIR                    sirurilor.
                SET     1,(FLAGS)       Semnal 'folosire imprimantă'
```

```
                CALL  0EDF,CLEAR-PRB      si se sterge bufferul
                                          imprimantei.
                LD    (DF-SZ),+02         Se setează mărimea părtii
                CALL  0D6B,CLS            inferioare a ecranului si se
                                          sterge tot ecranul.
                XOR   A                   Acum se tipăreste mesajul
                LD    DE,+1538            '@ 1982 Sinclair Research
                CALL  0C0A,PO-MSG         Ltd' pe ultima linie.
                SET   5,(TV-FLAG)         Semnal 'se cere stergerea
                                          părtii inferioare'.
                JR    12A9,MAIN-1         Salt înainte în bucla
                                          principală de executie.
```

THE 'MAIN EXECUTION' LOOP (BUCLA 'EXECUTIE PRINCIPALA')
Bucla executie principală se întinde de la locatia 12A2 pînă la locatia 15AE si ea controlează 'modul de editare', executia comenzilor directe si prezentarea activitătii.

```
12A2 MAIN-EXEC  LD    (DF-SZ),+02         Mărimea părtii inferioare a
                                          ecranului este de două linii.
                CALL  1795,AUTO-LIST      Se execută o listare
                                          automată.
12A9 MAIN-1     CALL  16B0,SET-MIN        Toate zonele de la E-LINE în
                                          continuare au minimul
                                          configuratiei dat.
12AC MAIN-2     LD    A,+00               Se deschide canalul 'K'
                CALL  1601,CHAN-OPEN      înainte de a se apela EDITOR.
                CALL  0F2C,EDITOR         EDITOR este apelat pentru a
                                          permite utilizatorului să
                                          construiască o linie BASIC.
                CALL  1B17,LINE-SCAN      Linia curentă este scanată
                                          pentru corectarea sintaxei.
                BIT   7,(ERR-NR)          Salt înainte dacă sintaxa
                JR    NZ,12CF,MAIN-3      este corectă.
                BIT   4,(FLAGS2)          Salt înainte dacă s-a folosit
                JR    Z,1303,MAIN-4       alt canal decît canalul 'K'.
                LD    HL,(E-LINE)         Se indică începutul unei linii
                                          cu eroare.
                CALL  11A7,REMOVE-FP      Se mută din nou forma
                                          punctului de lansare din
                                          această linie.
                LD    (ERR-NR),+FF        Se resetează ERR-NR si salt
                JR    12AC,MAIN-2         înapoi la MAIN-2 lăsînd
                                          listarea neschimbată.
```

'Linia editată' a trecut partea de sintaxă si cele trei tipuri de linii care sînt posibile vor trebui să fie deosebite una de cealaltă.

```
12CF MAIN-3     LD    HL,(E-LINE)         Se indică începutul liniei.
                LD    (CH-ADD),HL         Se setează începutul lui
                                          CH-ADD.
                CALL  19FB,E-LINE-NO      Se aduce numărul oricărei
                LD    A,B                 linii în BC. Este numărul
                OR    C                   liniei unul valid?
                JR    NZ,155D,MAIN-ADD    Salt dacă este, si se adaugă
                                          noua linie programului
                                          existent.
                RST   0018                Se aduce primul caracter al
                CP    +0D                 liniei si se testează dacă
                JR    Z,12A2,MAIN-EXEC    linia este doar 'carriage
                                          return'. Dacă este, salt
                                          înapoi.
```

'Linia editată' trebuie să înceapă cu o comandă BASIC directă si această linie va fi prima linie ce trebuie interpretată.

```
                BIT   0,(FLAGS2)          Se sterge întregul ecran dacă
                CALL  NZ,0DAF,CL-ALL      nu fanioanele nu arată caă nu
                                          este necesar.
                CALL  0D6E,CLS-LOWER      Oricum se sterge partea de
                                          jos a ecranului.
                LD    A,+19               Se încarcă valoarea potrivită
                SUB   (S-POSN-hi)         pentru contor defilare.
                LD    (SCR-CT),A
                SET   7,(FLAGS)           Semnal ' executie linie'.
                LD    (ERR-NR),+FF        Asigurare că ERR-NR este
                                          corect.
                LD    (NSPPC),+01         Se prelucrează prima
                                          instructie din linie.
```

```
                CALL  1B8A,PROG-RUN       Acum se interpretează linia.
                                          Notă: Adresa 1303 este
                                          încărcată în stiva
                                          calculatorului si este
                                          adresată cu ERR-SP.
```

După ce linia a fost interpretată si toate actiunile consecutive au fost executate se execută o revenire în MAIN-4, asa că se poate face o prezentare.

```
1303 MAIN-4     HALT                      Se activează întreruperea
                                          mascabilă.
                RES   5,(FLAGS)           Semnal 'gata pentru o tastă
                                          nouă'.
                BIT   1,(FLAGS2)          Se goleste bufferul
                CALL  NZ,0ECD,COPY-BUFF   imprimantei dacă a fost
                                          folosit.
                LD    A,(ERR-NR)          Se aduce numărul de eroare si
                INC   A                   se incrementează.
1313 MAIN-6     PUSH  AF                  Se salvează noua valoare.
                LD    HL,+0000            Variabilele sistem FLAGX,
                LD    (FLAGX),H           X-PTR-hi & DEFADD sînt puse
                LD    (X-PTR-hi),H        toate zero.
                LD    (DEFADD),HL
                LD    HL,+0001            Se asigură că sirul +00
                LD    (STRMS-6),HL        indică canalul 'K'.
                CALL  16B0,SET-MIN        Se sterg toate spatiile de
                                          lucru si stiva
                                          calculatorului.
                RES   5,(FLAGX)           Semna 'mod editare'.
                CALL  0D6E,CLS-LOWER      Se sterge partea inferioară
                                          a ecranului.
                SET   5,(TV-FLAG)         Semnal 'partea inferioară a
                                          ecranului cere stergere'.
                POP   AF                  Se aduce valoarea reportată.
                LD    B,A                 Se face o copie în B.
                CP    +0A                 Salt înainte cu prezentarea
                JR    C,133C,MAIN-5       numerelor '0 la 9'.
                ADD   A,+07               Se adună valorile ASCII ale
                                          literelor deplasamabtului.
133C MAIN-5     CALL  15EF,OUT-CODE       Se tipăreste codul prezentat
                LD    A,+20               urmat de un 'spatiu'.
                RST   0010,PRINT-A-1
                LD    A,B                 Se aduce valoarea prezentată
                LD    DE,+1391            si se foloseste pentru a
                CALL  0C0A,PO-MSG         identifica mesajul prezentat
                                          cerut.
                XOR   A                   Se tipăreste mesajul care
                LD    DE,+1536            este urmat de o 'virgulă' si
                CALL  0C0A,PO-MSG         de un spatiu'.
                LD    BC,(PPC)            Acum se aduce numărul liniei
                CALL  1A1B,OUT-NUM1       curente si se tipăreste.
                LD    A,+3A               El va fi urmat de ':'.
                RST   0010,PRINT-A-1
                LD    C,(SUBPPC)          Se aduce numărul instructiei
                LD    B,+00               curente în registrul pereche
                CALL  1A1B,OUT-NUM1       BC si se tipăreste.
                CALL  1097,CLEAR-SP       Se sterge spatiul de editare.
                LD    A,(ERR-NR)          Se aduce din nou numărul de
                                          eroare.
                INC   A                   Se incrementează ca de obicei
                JR    Z,1386,MAIN-9       Dacă programul a reusit nu
                                          mai există 'continuare', deci
                                          se execută salt.
                CP    +09                 Dacă programul se opreste cu
                JR    Z,1373,MAIN-6       'instructia STOP' sau 'BREAK
                CP    +15                 în program', continuarea se
                JR    NZ,1376,MAIN-7      face de la următoarea
1373 MAIN-6     INC   (SUBPPC)            instructiune; altfel SUBPPC
                                          este neschimbat.
1376 MAIN-7     LD    BC,+0003            Variabilele sistem OLDPPC &
                LD    DE,+5C70            OSPPC sînt făcute să conțină
                                          linia de continuare si
                                          numerele instructiei.
                LD    HL,+5C44            Valorile folosite sînt acelea
                BIT   7,(NSPPC)           care în PPC & SUBPPC, dar nu
                JR    Z,1384,MAIN-8       în NSPPC, arată că 'break'
                ADD   HL,BC               este necesar înainte de
                                          'jump' (salt).
1384 MAIN-8     LDDR                      (de exemplu, după o
```

```
                                                    instructie GO TO, etc.)
1386 MAIN-9     LD    (NSPPC),+FF                   Se resetează NSPPC pentru a
                                                    indica 'nu este salt'.
                RES   3,(FLAGS)                     Se selectează 'modul K'.
                JP    12AC,MAIN-2                   In final se execută salt
                                                    înapoi, dar nu apare listarea
                                                    programului pînă ce nu este
                                                    cerută.
```

THE REPORT MESSAGES (PREZENTAREA MESAJELOR)
Fiecare mesaj este dat cu ultimul caracter invertit (+80 hexa).

```
1391 DEFB +80       - trecere peste octetul initial
1392 Report 0       - 'OK'
1394 Report 1       - 'NEXT fără FOR'
13A4 Report 2       - 'Nu se găseste variabila'
13B6 Report 3       - 'Subscriere gresită'
13C6 Report 4       - 'Afară din memorie'
13D2 Report 5       - 'Afară din ecran'
13DF Report 6       - 'Număr prea mare'
13ED Report 7       - 'RETURN fără GOSUB'
1401 Report 8       - 'Sfîrsitul fisierului'
140C Report 9       - 'Instructiunea STOP'
141A Report A       - 'Argument invalid'
142A Report B       - 'Intreg afară din rang'
143E Report C       - 'Fără sens în BASIC'
144F Report D       - 'Repetare BREAK-CONT'
1463 Report E       - 'Afară din DATA'
146E Report F       - 'Nume fisier invalid'
147F Report G       - 'Nu mai sînt locuri pentru linie'
148F Report H       - 'STOP în INPUT'
149C Report I       - 'FOR fără NEXT'
14AC Report J       - 'Unitate I/O invalidată'
14BE Report K       - 'Culoare invalidată'
14CC Report L       - 'BREAK în program'
14DE Report M       - 'Nu este bun RAMTOP'
14EC Report N       - 'Instructie pierdută'
14FA Report O       - 'Sir invalidat'
1508 Report P       - 'FN fără DEF'
1516 Report Q       - 'Eroare parametru'
1525 Report R       - 'Eroare încărcare bandă'
```

Mai există următoarele două mesaje.

```
1537 ', '           - ca o 'virgulă' si un 'spatiu'
1539 '© 1982 Sinclair Research Ltd'
```

Prezentarea G - nu mai sînt locuri pentru linie.

```
1555 REPORT-G   LD    A,+10                         'G' are codul '10+07+30'.
                LD    BC,+0000                      Se sterge BC.
                JP    1313,MAIN-G                   Salt înapoi pentru prezentare
```

THE 'MAIN-ADD' SUBROUTINE
Această subrutină permite unei noi linii BASIC să fie adăugată unui program BASIC existent, în spatiul programului. Dacă o linie are atît o versiune veche cît si una nouă, atunci cea veche se 'reface'. O linie nouă care contine doar un număr de linie nu va trimisă în zona programului.

```
155D MAIN-ADD   LD    (E-PPC),BC                    Se face noul număr de linie
                                                    linie 'curentă.
                LD    HL,(CH-ADD)                   Se aduce CH-ADD si se
                EX    DE,HL                         salvează în DE.
                LD    HL,+1555                      Se salvează adresa lui
                PUSH  HL                            REPORT-G în stiva
                                                    calculatorului.
                LD    HL,(WORKSP)                   Se aduce WORKSP.
                SCF                                 Se determină lungimea liniei
                SBC   HL,DE                         de după numărul liniei care
                                                    a avut 'carriage return'.
                PUSH  HL                            Se salvează lungimea.
                LD    H,B                           Se mută numărul liniei în
                LD    L,C                           registrul pereche HL.
                CALL  196E,LINE-ADDR                Există o linie care are acest
                                                    număr?
                JR    NZ,157D,MAIN-ADD1             Salt dacă nu există.
                CALL  1988,NEXT-ONE                 Se caută lungimea liniei
                CALL  19E8,RECLAIM-2                'vechi' si se reface.
157D MAIN-ADD1  POP   BC                            Se aduce lungimea liniei
```

```
                LD    A,C               'noi'si salt înainte dacă
                DEC   A                 este numai 'număr de linie
                OR    B                 si carriage return'.
                JR    15AB,MAIN-ADD2
                PUSH  BC                Se salvează lungimea.
                INC   BC                Sînt necesare patru locatii
                INC   BC                suplimentare.
                INC   BC                De exemplu, două pentru
                INC   BC                număr & două pentru lungime.
                DEC   HL                HL va indica locatia dinainte
                                        de 'destinatie'.
                LD    DE,(PROG)         Se salvează valoarea curentă
                PUSH  DE                a lui PROG pentru a preveni
                                        deteriorarea cînd se adaugă
                                        prima linie.
                CALL  1655,MAKE-ROOM    Se creează spatiu pentru noua
                                        linie.
                POP   HL                Se aduce vechea valoare a lui
                LD    (PROG),HL         PROG si se reface.
                POP   BC                Se face o copie a lungimii
                PUSH  BC                liniei (fără parametrii).
                INC   DE                DE va indica locatia finală
                                        a noului spatiu si HL va
                LD    HL,(WORKSP)       indica 'carriage return' al
                DEC   HL                unei linii noi din spatiul de
                DEC   HL                editare.
                LDDR                    Acum se copiază linia.
                LD    HL,(E-PPC)        Se aduce numărul liniei.
                EX    DE,HL             Destinatia se găseste în HL
                                        & numărul în DE.
                POP   BC                Se aduce noua lungime a
                                        liniei.
                LD    (HL),B            Octetul superior al lungimii.
                DEC   HL
                LD    (HL),C            Octetul inferior al lungimii.
                DEC   HL
                LD    (HL),E            Octetul inferior al numărului
                                        liniei.
                DEC   HL
                LD    (HL),D            Octetul superior al numărului
                                        liniei.
15AB MAIN-ADD2  POP   AF                Salvare adresă REPORT-G.
                JP    12A2,MAIN-EXEC    Salt înapoi si de data asta
                                        se execută o listare
                                        automată.
```

THE 'INITIAL CHANNEL INFORMATION'
Initial sînt patru canale - 'K', 'S', 'R' & 'P' - pentru a comunica cu 'tastatura', 'ecranul', 'spatiul de lucru' si 'imprimanta'. Pentru fiecare canal adresa rutinei de iesire vine după adresa rutinei de intrare si codul canalului.

```
15AF DEFB    F4  09  -  PRINT-OUT
     DEFB    A8  10  -  KEY-INPUT
     DEFB    4B      -     'K'
15B4 DEFB    F4  09  -  PRINT-OUT
     DEFB    C4  15  -  REPORT-J
     DEFB    53      -     'S'
15B9 DEFB    81  0F  -  ADD-CHAR
     DEFB    C4  15  -  REPORT-J
     DEFB    52      -     'R'
15BE DEFB    F4  09  -  PRINT-OUT
     DEFB    C4  15  -  REPORT-J
     DEFB    50      -     'P'
15C3 DEFB    80      - Indicator sfîrsit.
```

Prezentare J - unitate I/O invalidată.

```
15C4 REPORT-J   RST   0008,ERROR-1      Se apelează rutina de tratare
                DEFB  +12               eroare.
```

THE 'INITIAL STREAM DATA' ('SIRUL INITIAL DE INFORMATII')

```
15C6 DEFB    01  00  - sirul +FD conduce la canalul 'K'
15C8 DEFB    06  00  - sirul +FE conduce la canalul 'S'
15CA DEFB    0B  00  - sirul +FF conduce la canalul 'R'
15CC DEFB    01  00  - sirul +00 conduce la canalul 'K'
15CE DEFB    01  00  - sirul +01 conduce la canalul 'K'
15D0 DEFB    06  00  - sirul +02 conduce la canalul 'S'
```

15D2 DEFB 10 00 - sirul +03 conduce la canalul 'P'

THE 'WAIT-KEY' SUBROUTINE (SUBRUTINA 'ASTEPTARE TASTA')
Această subrutină este o subrutină de control pentru apelarea subrutinei curente de intrare.

```
15D4 WAIT-KEY     BIT   5,(TV-FLAG)          Salt înainte dacă fanioanele
                  JR    NZ,15DE,WAIT-KEY1    indică faptul că partea de
                                             jos a ecranului nu a cerut
                                             stergere.
                  SET   3,(TV-FLAG)          Altfel se semnalează 'se
                                             consideră că s-a schimbat
                                             modul'.
15DE WAIT-KEY1    CALL  15E6,INPUT-AD        Se apelează indirect rutina
                                             de intrare, prin INPUT-AD.
                  RET   C                    Revenire.
                  JR    Z,15DE,WAIT-KEY1     'Dacă nici o tastă nu a fost
                                             apăsată' se resetează
                                             fanioanele de transport si
                                             de zero; altfel se semnalează
                                             eroare.
```

Prezentarea 8 - Sfîrsitul fisierului

```
15E4 REPORT-8     RST   0008,ERROR-1         Se apelează rutina de tratare
                  DEFB  +07                  eroare.
```

THE 'INPUT-AD' SUBROUTINE (SUBRUTINA 'INTRODUCERE ADRESA')
Registrii sînt salvati si HL va indica adresa de intrare.

```
15E6 INPUT-AD     EXX
                  PUSH  HL
                  LD    HL,(CURCHL)


                  INC   HL
                  INC   HL
                  JR    15F7,CALL-SUB
```

THE 'MAIN PRINTING' SUBROUTINE
Această subrutină este apelată sau printr-o valoare absolută sau prin codul propriu de caracter în registrul A.

```
15EF OUT-CODE     LD    E,+30                Se creste valoarea din
                  ADD   A,E                  registrul A cu +30.
15F2 PRINT-A-2    EXX                        Se salvează din nou registrii
                  PUSH  HL
                  LD    HL,(CURCHL)          Se aduce adresa de bază a
                                             canalului curent. Aceasta va
                                             indica o adresa de iesire.
```

Acum se apelează subrutina actuală. HL va indica adresa de iesire sau de intrare, după cum se indică.

```
15F7 CALL-SUB     LD    E,(HL)               Se aduce octetul inferior.
                  INC   HL
                  LD    D,(HL)               Se aduce octetul superior.
                  EX    DE,HL                Se mută adresa în registrul
                                             pereche HL.
                  CALL  162C,CALL-JUMP       Se cheamă subrutina actuală.
                  POP   HL                   Se refac registrii.
                  EXX
                  RET                        Se revine dacă nu a
                                             intervenit o eroare.
```

THE 'CHAN-OPEN' SUBROUTINE (SUBRUTINA 'DESCHIDERE CANAL')
Subrutina se apelează cu registrul A continînd un număr valid de sir - în mod normal de la +FD la +03. Apoi depinde de sirul de informatii dacă un canal anumit va fi făcut canal curent.

```
1601 CHAN-OPEN    ADD   A,A                  Valoarea din registrul A este
                  ADD   A,+16                dublată si apoi mărită cu +16
                  LD    L,A                  Rezultatul se mută în L.
                  LD    H,+5C                Adresa 5C16 este adresa de
                                             bază a sirului +00.
                  LD    E,(HL)               Se aduce primul octet al
                  INC   HL                   sirului de informatii cerut;
                  LD    D,(HL)               apoi al doilea bit.
                  LD    A,D                  Va apare eroare dacă ambii
```

```
                    OR    E                       octeti sînt zero; altfel
                    JR    NZ,1610,CHAN-OP-1       salt înainte.
```

Prezentare 0 - Sir invalid

```
160E REPORT-0       RST   0008,ERROR-1            Se apelează rutina de tratare
                    DEFB  +17                     eroare.
```

Folosind acum sirul de informatii se găseste adresa de bază a canalului de informatii asociat acestui sir.

```
1610 CHAN-OP-1      DEC   DE                      Se reduce sirul de informatii
                    LD    HL,(CHANS)              Adresa de bază pentru toată
                                                  zona canalului de informatii.
                    ADD   HL,DE                   Se formează adresa cerută în
                                                  acest spatiu.
```

THE 'CHAN-FLAG' SUBROUTINE
Fanioanele corespunzătoare pentru diferitele canale sînt setate de această subrutină.

```
1615 CHAN-FLAG      LD    (CURCHL),HL             Registrul pereche HL contine
                                                  adresa de bază a unui anumit
                                                  canal.
                    RES   4,(FLAGS2)              Semnal 'se foloseste alt
                                                  canal decît canalul K'.
                    INC   HL                      Se trece peste adresele de
                    INC   HL                      de iesire si intrare si HL va
                    INC   HL                      indica codul canalului.
                    INC   HL
                    LD    C,(HL)                  Se aduce codul.
                    LD    HL,+162D                Adresa de bază a 'tabelului
                                                  de trecere cod canal'.
                    CALL  16DC,INDEXER            Se indexează în acest tabel
                                                  si se localează deplasamentu
                    RET   NC                      cerut; se revine dacă nu s-
                                                  găsit un cod canal identic.
                    LD    D,+00                   Se trece deplasamentul în
                    LD    E,(HL)                  registrul pereche DE.
                    ADD   HL,DE                   Salt înainte la rutina
1620 CALL-JUMP      JP    (HL)                    corespunzătoare pentru
                                                  setarea fanioanelor.
```

THE 'CHANNEL CODE LOOK-UP' TABLE

```
162D  DEFB  4B  06  - canal 'K'   deplasament +06   adresa 1634
162F  DEFB  53  12  - canal 'S'   deplasament +12   adresa 1642
1631  DEFB  50  1B  - canal 'P'   deplasament +1B   adresa 164D
1633  DEFB  00      - marcator
                      sfîrsit
```

THE 'CHANNEL 'K' FLAG' SUBROUTINE (SUBRUTINA 'FANION CANAL 'K'')

```
1634 CHAN-K         SET   0,(TV-FLAG)             Semnal 'se foloseste partea
                                                  inferioară a ecranului'.
                    RES   5,(FLAGS)               Semnal 'pregătit pentru o
                                                  tastă'.
                    SET   4,(FLAGS2)              Semnal 'se foloseste canalul
                                                  'K''.
                    JR    1646,CHAN-S-1           Salt înainte.
```

THE 'CHANNEL 'S' FLAG' SUBROUTINE (SUBRUTINA 'FANION CANAL 'S'')

```
1642 CHAN-S         RES   0,(TV-FLAG)             Semnal 'se foloseste partea
                                                  principală a ecranului'.
1646 CHAN-S-1       RES   1,(FLAGS)               Semnal 'imprimanta nu a fost
                                                  folosită'.
                    JP    0D4D,TEMPS              Iesire prin TEMPS asa încît
                                                  să se seteze variabilele
                                                  sistem culoare.
```

THE 'CHANNEL 'P' FLAG' SUBROUTINE (SUBRUTINA 'FANION CANAL 'P'')

```
164D CHAN-P         SET   1,(FLAGS)               Semnal 'imprimanta în
                    RET                           functiune'.
```

THE 'MAKE-ROOM' SUBROUTINE (SUBRUTINA 'FACE LOC')
Aceasta este o subrutină foarte importantă. Ea este apelată in multe ocazii

pentru a 'lărgi' o zonă. În toate cazurile registrul pereche HL indică locatia de după locul unde 'camera' este cerută si registrul pereche BC continea lungimea 'camerei' de care este nevoie.
Cînd se cere un singur spatiu, subrutina este introdusă prin ONE-SPACE.

```
1652 ONE-SPACE      LD    BC,+0001          Se cere doar o singură
                                            locatie aditională.
1655 MAKE-ROOM      PUSH  HL                Salvare indicator.
                    CALL  1F05,TEST-ROOM    Se asigură că este accesibilă
                                            suficientă memorie pentru ca
                                            taskul să fi preluat.
                    POP   HL                Refacere indicator.
                    CALL  1664,POINTERS     Se schimbă toti indicatorii
                                            înainte de a realiza 'camera'
                                            (locatia).
                    LD    HL,(STKEND)       HL contine noul STKEND.
                    EX    DE,HL             Schimbare 'vechi' si 'nou'.
                    LDDR                    Acum se realizează 'locatia'
                    RET                     si revenire.
```

Notă: Această subrutină revine cu registrul pereche HL inidicînd locatia dinaintea noii 'camere' si registrul pereche DE indicînd sfîrsitul noii locatii. De aceea, noua 'cameră' are descrierea de la '(HL)+1' la '(DE)+1' inclusiv.
După cum 'noile locati' retin 'vechile lor valori' este de asemenea posibil să se considere noua 'cameră' ca fiind făcută după locatia originală '(HL)' si astfel va avea descrierea de la '(HL)+2' la (DE)+1'.
De fapt programatorul pare să aibe o preferintă pentru a 'doua descriere' si acest fapt poate duce la confuzii.

THE 'POINTERS' SUBROUTINE (SUBRUTINA 'INDICATORI')
Intotdeauna cînd o zonă trebuie 'făcută' sau 'refăcută' variabilele sistem care adresează locatia dincolo de 'pozitia' schimbării trebuie modificată după cum se cere. La intrare, în registrul pereche BC se află numărul octetilor indusi, iar în registrul pereche HL se află adresele locatiei dinaintea 'pozitiei'.

```
1664 POINTERS       PUSH  AF                Se salvează registrii.
                    PUSH  HL                Se copiază adresa 'pozitiei'.
                    LD    HL,+5C4B          Aceasta este adresa lui VARS,
                    LD    A,+0E             primul dintre cei 14
                                            indicatori sistem.
```

Acum se intră într-o buclă în care se va lua în considerare fiecare indicator pe rînd. Se vor schimba doar acei indicatori care arată dincolo de 'pozitie'.

```
166B PTR-NEXT       LD    E,(HL)            Se aduc cei doi octeti ai
                    INC   HL                indicatorului curent.
                    LD    D,(HL)
                    EX    (SP),HL           Se schimbă variabila sistem
                                            cu adresa 'pozitiei'.
                    AND   A                 Fanionul de transport va fi
                    SBC   HL,DE             setat dacă adresa variabilei
                    ADD   HL,DE             sistemului trebuie mărită.
                    EX    (SP),HL           Se reface 'pozitia'.
                    JR    NC,167F,PTR-DONE  Salt înainte dacă
                                            indicatorul trebuie să fie la
                                            stînga; altfel se schimbă.
                    PUSH  DE                Se salvează valoarea veche.
                    EX    DE,HL             Acum se adună la valoarea din
                    ADD   HL,BC             BC vechea valoare.
                    EX    DE,HL
                    LD    (HL),D            Se introduce noua valoare în
                    DEC   HL                variabila sistem - octetul
                    LD    (HL),E            superior înaintea octetului
                                            inferior.
                    INC   HL                Se indică octetul superior.
                    POP   DE                Se aduce vechea valoare.
167F PTR-DONE       INC   HL                Se indică următoarea
                    DEC   A                 variabilă sistem si salt
                    JR    NZ,166B,PTR-NEXT  înapoi pînă cînd toti cei
                                            14 indicatori au fost
                                            considerati.
```

Acum se află marimea blocului ce trebuie mutat.

```
                    EX    DE,HL             Se pune vechea valoare a lui
                    POP   DE                STKEND în HL si se refac
                    POP   AF                ceilalti registri.
```

```
                AND   A                 Acum se determină diferența
                SBC   HL,DE             între vechea valoare a lui
                LD    B,H               STKEND si 'poziție'.
                LD    C,L               Rezultatul se transferă în BC
                INC   BC                si se adună '1' pentru octet
                                        inclusiv.
                ADD   HL,DE             Reformarea vechii valori a
                EX    DE,HL             lui STKEND si trecrea ei în
                RET                     DE înainte de revenire.
```

THE 'COLLECT A LINE NUMBER' SUBROUTINE (SUBRUTINA 'COLECTARE NUMAR DE LINIE')
La intrare, registrul pereche HL va indica locatia luată în considerare. Dacă locatia contine o valoare care reprezintă partea superioară a unui octet care convine pentru un număr de linie, atunci numărul de linie este trecut în DE. După cum, dacă nu este asa, locatia adresată prin DE este testată pe loc; si dacă si acest test este nereusit, atunci se reface numărul de linie zero.

```
168F LINE-ZERO  DEFB  +00              Număr de linie zero.
                DEFB  +00
1691 LINE-NO-A  EX    DE,HL             Se consideră celălalt
                                        indicator.
                LD    DE,+168F          Se foloseste numărul de
                                        linie zero.
```

In mod normal, punctul de intrare este LINE-NO.

```
1695 LINE-NO    LD    A,(HL)            Se aduce octetul superior si
                AND   +C0               se testează.
                JR    NZ,1691,LINE-NO-A Salt înapoi dacă nu este
                                        convenabil.
                LD    D,(HL)            Se aduce octetul inferior si
                INC   HL                se revine.
                LD    E,(HL)
                RET
```

THE 'RESERVE' SUBROUTINE (SUBRUTINA 'REZERVARE')
In mod normal, această subrutină este apelată folosind RST 0030, BC-SPACES.
La intrare, ultima valoare din stiva calculatorului este WORKSP si valoarea de după ea este numărul spatiilor care trebuie 'rezervate'.
Această subrutină întotdeauna realizează 'camere' între spatiul de lucru existent si stiva calculatorului.

```
169E RESERVE    LD    HL,(STKBOT)       Se aduce valoarea curentă a
                DEC   HL                lui STKBOT si se
                                        decrementează pentru a
                                        obtine ultima locatie a
                                        spatiului de lucru.
                CALL  1655,MAKE-ROOM    Se fac 'BC spatii'.
                INC   HL                Se indică primul spatiu nou
                INC   HL                si apoi al doilea.
                POP   BC                Se aduce vechea valoarea a
                LD    (WORKSP),BC       lui WORKSP si se reface.
                POP   BC                Se reface BC - numărul de
                                        spatii.
                EX    DE,HL             Schimbare indicatori.
                INC   HL                HL indică primul octet
                                        înlocuit.
                RET                     Revenire.
```

Notă: Se poate lua de asemenea în considerare că subrutina execută revenirea cu registrul pereche DE indicînd 'primul octet aditional' si cu registrul pereche HL indicînd 'ultimul octet aditional', acesti octeti aditionali fiind adăugati după locatia originară '(HL)+1'.

THE 'SET-MIN' SUBROUTINE
Această subrutină resetează zona de editare si zonele de după acesta la mărimea lor minimă. De fapt, ea 'sterge' zonele.

```
16B0 SET-MIN    LD    HL,(E-LINE)       Se aduce continutul E-LINE.
                LD    (HL),+0D          Zona de editare contine doar
                LD    (K-CUR),HL        caracterul 'carriage return'
                INC   HL                si marcatorul de sfîrsit.
                LD    (HL),+80
                INC   HL                Avans pentru curătirea
                LD    (WORKSP),HL       spatiului de lucru.
```

Această intrare 'va curăta' spatiul de lucru si stiva calculatorului.

```
16BF SET-WORK   LD    HL,(WORKSP)       Se aduce continutul lui
```

```
                                                    WORKSP.
                    LD      (STKBOT),HL             Aceasta curàtà stiva.
```

Aceastà intrare 'va curàta' numai stiva calculatorului.

```
16C5 SET-STK        LD      HL,(STKBOT)             Se aduce continutul lui
                                                    STKBOT.
                    LD      (STKEND),HL             Aceasta curàtà stiva.
```

In toate aceste cazuri MEM adreseazà zona de memorie a calculatorului.

```
                    PUSH    HL                      Se salveazà STKEND.
                    LD      HL,+5C92                Baza zonei de memorie.
                    LD      (MEM),HL                Se seteazà MEM de la acestà
                                                    adresà.
                    POP     HL                      Se reface STKEND în registrul
                    RET                             pereche HL înainte de a se
                                                    reveni.
```

THE 'RECLAIM THE EDIT-LINE' SUBROUTINE (SUBRUTINA 'REFACERE LINIE EDITARE')

```
16D4 REC-EDIT       LD      DE,(E-LINE)             Se aduce continutul lui
                                                    E-LINE.
                    JP      19E5,RECLAIM-1          Refacere memorie.
```

THE 'INDEXER' SUBROUTINE (SUBRUTINA 'INDEXARE')
Aceastà subrutinà este folosità de mai multe ori pentru a càuta prin tabele. Intrarea este la INDEXER.

```
16DB INDEXER-1      INC     HL                      Avans pentru considerarea
                                                    urmàtoarelor perechi de
                                                    intràri.
16DC INDEXER        LD      A,(HL)                  Se aduce prima dintr-o
                    AND     A                       pereche de intràri, dar se
                    RET     Z                       revine dacà ea este zero -
                                                    marcatorul de sfîrsit.
                    CP      C                       Se comparà cu codul înlocuit.
                    INC     HL                      Se indicà a doua intrare.
                    JR      NZ,16DB,INDEXER-1       Salt înapoi dacà nu s-a gàsit
                                                    intrarea corectà.
                    SCF                             Fanionul de transport este
                    RET                             setat dupà o càutare reusità.
```

THE 'CLOSE#' COMMAND ROUTINE (RUTINA DE COMANDA 'CLOSE#' (închidere#))
Aceastà comandà permite utilizatorului sà închidà (CLOSE) siruri. Pentru siruri +00 la +03 informatia sir 'initialà' este refàcutà si aceste siruri nu pot fi închise.

```
16E5 CLOSE          CALL    171E,STR-DATA           Se aduce informatia existentà
                                                    pentru sir.
                    CALL    1701,CLOSE-2            Se verificà codul în acest
                                                    canal al sirului.
                    LD      BC,+0000                Pregàtire pentru punerea
                                                    informatiei sirului pe zero.
                    LD      DE,+A3E2                Pregàtire pentru a identifica
                    EX      DE,HL                   utilizarea sirurilor de la
                                                    +00 la +03.
                    ADD     HL,DE                   Fanionul de transport se
                                                    seteazà pentru siruri de la
                                                    +04 la +0F.
                    JR      C,16FC,CLOSE-1          Salt înainte cu aceste siruri
                    LD      BC,+15D4                în caz contrar se aflà
                    ADD     HL,BC                   intrarea corectà în tabelul
                                                    'informatie sir 'initialà'.
                    LD      C,(HL)                  Se aduce informatia initialà
                    INC     HL                      pentru sirurile de la +00 la
                    LD      B,(HL)                  +03.
16FC CLOSE-1        EX      DE,HL                   Acum se introduce informatia;
                    LD      (HL),C                  sau zero & zero sau valoarea
                    INC     HL                      initialà.
                    LD      (HL),B
                    RET
```

THE 'CLOSE-2' SUBROUTINE
Codul canalului asociat sirului care a fost închis trebuie sà fie 'K, 'S' sau 'P'.

```
1701 CLOSE-2        PUSH    HL                      Se salveazà adresa
                                                    informatiei sir.
```

```
                LD    HL,(CHANS)        Se aduce adresa de bază a
                ADD   HL,BC             zonei canalului de informatie
                                        si se caută informatia de
                                        canal pentru sirul care a
                                        fost închis.
                INC   HL                Se trece peste adresele
                INC   HL                subrutinei si se selectează
                INC   HL                codul pentru acel canal.
                LD    C,(HL)
                EX    DE,HL             Salvare indicator.
                LD    HL,+1716          Adresa de bază pentru tabelul
                                        'trecere prin sir închisă'.
                CALL  16DC,INDEXER      Se indexează în acest tabel
                                        si se locatează deplasamentul
                                        cerut.
                LD    C,(HL)            Se trece deplsamentul în
                LD    B,+00             registrul pereche BC.
                ADD   HL,BC             Salt înainte la rutina
                JP    (HL)              corespunzătoare.
```

THE 'CLOSE STREAM LOOK-UP' TABLE (TABELUL 'TRECERE PRIN SIR INCHIS')

```
1716  DEFB  4B 05  -  canal 'K'   deplasament +05,   adresa 171C
1718  DEFB  53 03  -  canal 'S'   deplasament +03,   adresa 171C
171A  DEFB  50 01  -  canal 'P'   deplasament +01,   adresa 171C
```

Notă: Nu este nici un marcator de sfîrsit la sfîrsitul acestui tabel.

THE 'CLOSE STREAM' SUBROUTINE (SUBRUTINA 'INCHIDE SIR')

```
171C CLOSE-STR  POP   HL                Se aduce indicatorul
                RET                     informatiei canal si se
                                        revine.
```

THE 'STREAM DATA' SUBROUTINE (SUBRUTINA 'INFORMATII SIR')
Această subrutină returnează în registrul pereche HL informatia sir pentru un sir dat.

```
171E STR-DATA   CALL  1E94,STK-TO-A     Numărul sirului dat este scos
                                        din stiva calculatorului.
                CP    +10               Va apare eroare dacă numărul
                JR    C,1727,STR-DATA1  sirului este mai mare decît
                                        +0F.
```

Prezentarea 0 - Sir invalid

```
1725 REPORT-0           0008,ERROR-1    Se apelează rutina de tratare
                DEFB  +17               eroare.
```

Se continuă cu numere de sir valide.

```
1727 STR-DATA1  ADD   A,+03             Acum se aranjează de la +03
                RLCA                    la +12, apoi de la +06 la
                                        +24.
                LD    HL,+5C10          Adresa de bază a spatiului
                                        informatiilor sir.
                LD    C,A               Se mută informatia sir în
                LD    B,+00             registrul pereche BC.
                ADD   HL,BC             Se indexează în spatiul
                LD    C,(HL)            informatiilor si se aduc doi
                INC   HL                octeti de informatii în
                LD    B,(HL)            registrul pereche BC.
                DEC   HL                Indicatorul adresează primul
                RET                     octet de informatii înaintea
                                        revenirii.
```

THE 'OPEN#' COMMAND ROUTINE (RUTINA DE COMANDA 'OPEN#' (deschidere))
Această comandă permite utilizatorului să deschidă siruri. Un cod canal trebuie înlocuit si el trebuie să fie 'K', 'k', 'S', 's', 'P' sau 'p'.
De notat că nu se face nici o încercare pentru a da sirurilor +00 la +03 informatiile lor initiale.

```
1736 OPEN       RST   0028,FP-CALC      Utilizare CALCULATOR.
                DEFB  +01,exchange      Se schimbă numărul de sir si
                DEFB  +38,end-calc      codul canalului.
                CALL  171E,STR-DATA     Se aduc informatii pentru
                                        sir.
                LD    A,B               Salt înainte dacă ambii
                OR    C                 octeti de informatii sînt
```

```
                JR    Z,1756,OPEN-1          zero, adicá sirul a fost
                                             într-o stare închisá.
                EX    DE,HL                  Se salveazá DE.
                LD    HL,(CHANS)             Se aduce CHANS - adresa de
                ADD   HL,BC                  bazá a canalului de
                INC   HL                     informatii si se gáseste
                INC   HL                     codul canalului asociat cu
                INC   HL                     sirul care a fost deschis.
                LD    A,(HL)
                EX    DE,HL                  Revenire DE.
                CP    +4B                    Codul adus din zona canalului
                JR    Z,1756,OPEN-1          de informatii trebuie sá fie
                CP    +53                    'K', 'S' sau 'P'; dacá nu
                JR    Z,1756,OPEN-1          este , va apare eroare.
                CP    +50
                JR    NZ,1725,REPORT-O
1756 OPEN-1     CALL  175D,OPEN-2            Se adunà în DE informatiile
                                             corespunzátoare.
                LD    (HL),E                 Informatiile se introduc în
                INC   HL                     doi octeti în zona de
                LD    (HL),D                 informatii a canalului.
                RET                          In final, revenire.
```

THE 'OPEN-2' SUBROUTINE
Octetii corespunzátori ai sirului de informatii pentru canalul asociat cu sirul care a fost deschis sînt gásiti.

```
175D OPEN-2     PUSH  HL                     Salvare HL.
                CALL  2BF1,STK-FETCH         Se aduc parametrii din
                                             canalul de cod.
                LD    A,B                    Va apare eroare dacá expresia
                OR    C                      înlocuitá este una invalidá;
                JR    NZ,1767,OPEN-3         aceasta este OPEN #5,"".
```

Prezentarea F - Nume fisier invalid

```
1765 REPORT-F   RST   0008,ERROR-1           Se apeleazá rutina de tratare
                DEFB  +0E                    eroare.
```

Se continuá dacá nu a intervenit nici o eroare.

```
1767 OPEN-3     PUSH  BC                     Salvare lungime expresie.
                LD    A,(DE)                 Se aduce primul caracter.
                AND   +DF                    Se face conversia codurilor
                                             pentru literele mici în
                                             codurile literelor mari.
                LD    C,A                    Se mutá codul în registrul C.
                LD    HL,+177A               Adresa de bazá a tabelului
                                             'trecere prin sir deschis'.
                CALL  16DC,INDEXER           Se indexeazá în acest tabel
                                             si se locatează deplasamentul
                                             cerut.
                JR    NC,1765,REPORT-F       Salt înapoi dacá nu a fost
                                             gásit.
                LD    C,(HL)                 Se trece deplasamentul în
                LD    B,+00                  registrul pereche BC.
                ADD   HL,BC                  HL va indica începutul
                                             subrutinei corespunzátoare.
                POP   BC                     Se aduce lungimea expresiei
                JP    (HL)                   înainte de a efectua saltul
                                             din subrutiná.
```

THE 'OPEN STREAM LOOK-UP' TABLE (TABELUL 'TRECERE PRIN SIR DESCHIS')

```
177A  DEFB  4B  06  - canal 'K',    deplasament +06,    adresa 1781
177C  DEFB  53  08  - canal 'S',    deplasament +08,    adresa 1785
177E  DEFB  50  0A  - canal 'P',    deplasament +0A,    adresa 1789
1780  DEFB  00      - marcator de sfîrsit.
```

THE 'OPEN-K' SUBROUTINE (SUBRUTINA 'DESCHIDE K')

```
1781 OPEN-K     LD    E,+01                  Octetii de informatii vor fi
                JR    1788,OPEN-END          +06 & +00.
```

THE 'OPEN-S' SUBROUTINE (SUBRUTINA 'DESCHIDE S')

```
1785 OPEN-S     LD    E,,+06                 Octetii de informatii vor fi
                JR    1788,OPEN-END          +06 & +00.
```

THE 'OPEN-P' SUBROUTINE (SUBRUTINA 'DESCHIDE P')

```
1789 OPEN-P       LD    E,+10              Octetii de informatii vor fi
                                           +10 & +00.
178B OPEN-END     DEC   BC                 Se decrementează lungimea
                  LD    A,B                expresiei si va apare eroare
                  OR    C                  dacă nu a fost un singur
                  JR    NZ,1765,REPORT-F   caracter; altfel se sterge
                  LD    D,A                registrul D, se reface HL
                  POP   HL                 si se execută revenire.
                  RET
```

THE 'CAT, ERASE, FORMAT & MOVE' COMMAND ROUTINES (RUTINELE DE COMANDA 'CAT (ridicare), ERASE (stergere), FORMAT (formatare), MOVE (mutare))
In sistemul standard SPECTRUM utilizarea acestor comenzi duce la executia prezentării 0 - Sir invalid.

```
1793 CAT-ETC      JR    1725,REPORT-0      Se dă această prezentare.
```

THE 'LIST & LLIST' COMMAND ROUTINES (RUTINELE DE COMANDA 'LIST & LLIST)
Rutinele din această parte de 16 K program sînt folosite pentru executarea listării programului programului BASIC curent. Fiecare linie trebuie să aibe numărul propriu de linie evaluat, dezvoltarea semnelor ei si cursorul corespunzător pozitionate.
Punctul de intrare AUTO-LIST este utilizat atît de rutina 'MAIN EXECUTION' (executie principală) cît si de editor, pentru executarea unei singure pagini de listing.

```
1795 AUTO-LIST    LD    (LIST-SP),SP       Indicatorul de stivă este
                                           salvat pentru a permite
                                           stivei calculatorului să fie
                                           resetată cînd se termină
                                           listarea (vezi PO-SCR, 0C55).
                  LD    (TV-FLAG),+10      Semnal 'listare automată în
                                           partea principală a
                                           ecranului'.
                  CALL  0DAF,CL-ALL        Se sterge această partea a
                                           ecranului.
                  SET   0,(FLAGS)          Schimbare în zona de editare.
                  LD    B,(DF-SZ)          Acum se sterge partea
                  CALL  0E44,CL-LINE       inferioară a ecranului.
                  RES   0,(FLAGS)          Se schimbă la loc.
                  SET   0,(FLAGS2)         Semnal 'ecran curat'.
                  LD    HL,(E-PPC)         Acum se aduce numărul liniei
                  LD    DE,(S-TOP)         'curente' si numărul liniei
                                           'automatic.
                  AND   A                  Dacă numărul 'curent' este
                  SBC   HL,DE              mai mic decît numărul
                  ADD   HL,DE              'automatic' atunci se execută
                  JR    C,17E1,AUTO-L-2    salt înainte pentru mărirea
                                           numărului 'automatic'.
```

Numărul 'automatic' trebuie modificat pentru a da o listare cu linia 'curentă' apărînd lîngă linia cea mai de jos a ecranului.

```
                  PUSH  DE                 Se salvează numărul
                                           'automatic'.
                  CALL  196E,LINE-ADDR     Se caută adresa începutului
                  LD    DE,+02C0           liniei 'curente'.
                  EX    DE,HL
                  SBC   HL,DE              Se salvează rezultatul în
                  EX    (SP),HL            stiva calculatorului pînă
                  CALL  196E,LINE-ADDR     cînd adresa liniei 'automatic
                                           va fi si ea găsită.
                  POP   BC                 Rezultatul se trece în
                                           registrul pereche BC.
```

Acum se introduce o buclă. Numărul liniei 'automatic' este incrementat la fiecare trecere pînă cînd este potrivit cu cel al liniei 'curente' ce va apărea la listare.

```
17CE AUTO-L-1     PUSH  BC                 Se salvează 'rezultatul'.
                  CALL  19B8,NEXT-ONE      Se caută adresa începutului
                                           liniei după linia 'automatic'
                                           prezentă (în DE).
                  POP   BC                 Se reface 'rezultatul'.
                  ADD   HL,BC              Se execută calculul si se
                  JR    C,17E4,AUTO-L-3    execută salt înainte cînd
                                           este gata.
```

```
                EX    DE,HL          Se mută adresa următoarei
                LD    D,(HL)         linii în registrul pereche HL
                INC   HL             si se adună numărul ei de
                LD    E,(HL)         linie.
                DEC   HL
                LD    (S-TOP),DE     Acum S-TOP se poate mări si
                JR    17CE,AUTO-L-1  se repetă testul cu noua
                                     linie.
```

Acum se poate executa listarea 'automatic'.

```
17E1 AUTO-L-2   LD    (S-TOP),HL     Cînd E-PPC este mai mic decît
                                     S-TOP.
17E4 AUTO-L-3   LD    HL,(S-TOP)     Se aduce numărul liniei din
                CALL  196E,LINE-ADDR vîrf si deci adresa ei.
                JR    Z,17ED,AUTO-L-4 Dacă linia nu poate fi găsită
                EX    DE,HL          se foloseste imediat DE.
17ED AUTO-L-4   CALL  1833,LIST-ALL  S-a executat listarea.
                RES   4,(TV-FLAG)    Revenirea se va face aici
                RET                  dacă nu este necesară
                                     defilarea pentru a arăta
                                     linia curentă.
```

THE 'LLIST' ENTRY POINT (PUNCTUL DE INTRARE 'LLIST')
Trebuie deschis canalul imprimantei.

```
17F5 LLIST      LD    A,+03          Se foloseste sirul +03.
                JR    17FB,LIST-1    Salt înainte.
```

THE 'LIST' ENTRY POINT (PUNCTUL DE INTRARE 'LIST')
Trebuie deschis canalul 'ecranului principal'.

```
17F9 LIST       LD    A,+02          Se foloseste sirul +02.
17FB LIST-1     LD    (TV-FLAG),+00  Se semnalează 'o listare
                                     obisnuită în partea de jos a
                                     ecranului'.
                CALL  2530,SYNTAX-Z  Se deschide canalul dacă nu
                CALL  NZ,1601,CHAN-OPEN este verificată sintaxa.
                RST   0018,GET-CHAR  Cu prezentul caracter în
                CALL  2070,STR-ALTER registrul A se verifică dacă
                                     sirul este schimbat.
                JR    C,181F,LIST-4  Salt înainte dacă nu s-a
                                     schimbat.
                RST   0018,GET-CHAR  Este caracterul prezent ';' ?
                CP    +3B
                JR    Z,1814,LIST-2  Dacă este, salt.
                CP    +2C            Este ',' ?
                JR    NZ,181A,LIST-3 Dacă nu este, salt.
1814 LIST-2     RST   0020,NEXT-CHAR Urmează o expresie numerică,
                CALL  1C82,EXPT-1NUM de exemple LIST #5.
                JR    1822,LIST-5    Salt înainte cu acesta.
181A LIST-3     CALL  1CE6,USE-ZERO  Altfel se foloseste zero si
                JR    1822,LIST-5    salt înainte.
```

Se ajunge în acest punct dacă sirul a fost neschimbat.

```
181F LIST-4     CALL  1CDE,FETCH-NUM Se aduce orice linie sau se
                                     foloseste zero dacă nu sînt
                                     înlocuiri.
1822 LIST-5     CALL  1BEE,CHECK-END Dacă s-a verificat sintaxa
                                     liniei de editare se face o
                                     mutare la următoarea
                                     instructiune.
                CALL  1E99,FIND-INT  Numărul liniei în BC.
                LD    A,B            Octetul superior în A.
                AND   +3F            Se limitează octetul superior
                LD    H,A            la ordinul corect si se trece
                LD    L,C            întregul număr al liniei în
                                     HL.
                LD    (E-PPC),HL     Se setează E-PPC si se
                CALL  196E,LINE-ADDR găseste adresa de început a
                                     acestei linii sau prima linie
                                     după ea actuala linie nu
                                     există.
                LD    E,+01          Fanion 'înaintea liniei
                                     curente'.
```

Acum este introdusă bucla pentru controlul tipăririi unei serii de linii.

```
1835 LIST-ALL     CALL  1855,OUT-LINE       Se tipăreste întreaga linie
                                            BASIC.
                  RST   0010,PRINT-A-1      Acesta va fi un 'carriage
                                            return'.
                  BIT   4,(TV-FLAG)         Salt înapoi dacă nu s-a
                  JR    Z,1835,LIST-ALL     lucrat cu listare automată.
                  LD    A,(DF-SZ)           De asemenea salt înapoi dacă
                  SUB   (S-POSN-hi)         mai există o parte din
                  JR    NZ,1835,LIST-ALL    ecranul principal care poate
                                            fi folosită.
                  XOR   E                   In acest punct se poate face
                  RET   Z                   o revenire dacă ecranul este
                                            plin si linia curentă a fost
                                            tipărită (E = +00).
                  PUSH  HL                  Oricum, dacă linia curentă
                  PUSH  DE                  lipseste din listare, atunci
                  LD    HL,+5C6C            S-TOP trebuie mărit si o
                  CALL  190F,LN-FETCH       linie următoare trebuie
                  POP   DE                  tipărită (folosind defilarea)
                  POP   HL
                  JR    1835,LIST-ALL
```

THE 'PRINT A WHOLE BASIC LINE' SUBROUTINE (SUBRUTINA 'TIPARIRE A UNEI INTREGI LINII BASIC')
Registrul pereche HL indică începutul liniei - locatia care contine octetul superior al numărului liniei.
Inainte de a se tipări numărul liniei el este testat pentru a se determina dacă vine înaintea liniei 'curente', este linia 'curentă' sau vine după linia 'curentă'.

```
1855 OUT-LINE     LD    BC,(E-PPC)          Se aduce numărul liniei
                  CALL  1980,CP-LINE        'curente' si se compară.
                  LD    D,+3E               Se încarcă registrul D cu
                                            cursorul liniei curente.
                  JR    Z,1865,OUT-LINE1    Salt înainte dacă se
                                            tipăreste linia 'curentă'.
                  LD    DE,+0000            Se încarcă registrul D cu
                                            zero (nu este cursorul) si se
                  RL    E                   setează registrul E încît să
                                            conțină +01 dacă linia este
                                            înainte de linia 'curentă' si
                                            +00 dacă este după ea.
                                            (Fanionul de transport vine
                                            din CP-LINES).
1865 OUT-LINE     LD    (BREG),E            Salvare marcator de linie.
                  LD    A,(HL)              Se aduce octetul superior al
                  CP    +40                 numărului liniei si se face o
                  POP   BC                  revenire completă dacă s-a
                  RET   NC                  terminat listarea.
                  PUSH  BC
                  CALL  1A28,OUT-NUM-2      Acum se poate lista numărul
                                            liniei - cu spatii de fond.
                  INC   HL                  Se mută indicatorul ca să
                  INC   HL                  adreseze primul cod de
                  INC   HL                  comandă în linie.
                  RES   0,(FLAGS)           Semnal 'permite spatii de
                                            fond'.
                  LD    A,D                 Se aduce codul cursorului
                  AND   A                   si salt înainte dacă cursorul
                  JR    Z,1881,OUT-LINE3    nu trebuie listat.
                  RST   0010,PRINT-A-1      Acum se tipăreste cursorul.
187D OUT-LINE2    SET   0,(FLAGS)           Semnal 'acum nu sînt spatii
                                            de fond.
1881 OUT-LINE3    PUSH  DE                  Se salvează registrii.
                  EX    DE,HL               Se mută indicatorul în DE.
                  RES   2,(FLAGS2)          Semnal 'nu între cote'.
                  LD    HL,+5C3B            Aceasta este adresa lui FLAGS
                  RES   2,(HL)              Semnal 'tipărire în mod K'.
                  BIT   5,(FLAGX)           Salt înainte dacă nu este mod
                  JR    Z,1894,OUT-LINE4    INPUT.
                  SET   2,(HL)              Semnal 'tipărire în mod L'.
```

Acum este introdusă o buclă care să tipărească toate codurile din restul liniei BASIC - sărind peste formele punctului flotant după caz.

```
1894 OUT-LINE4    LD    HL,(X-PTR)          Se aduce indicatorul erorii
                  AND   A                   de sintaxă si salt înainte
                  SBC   HL,DE               dacă nu este momentul pentru
                  JR    NZ,18A1,OUT-LINE5   tipărirea marcatorului de
```

```
                                                    eroare.
                LD    A,+3F                         Acum se tipăreste marcatorul
                                                    de eroare.
                CALL  18C1,OUT-FLASH                Pîlpîie '?'.
18A1 OUT-LINES  CALL  18E1,OUT-CURS                 Se consideră dacă este timpul
                                                    să se tipărească cursorul.
                EX    DE,HL                         Acum se mută indicatorul în
                                                    HL.
                LD    A,(HL)                        Se aduce fiecare caracter pe
                                                    rînd.
                CALL  18B6,NUMBER                   Dacă caracterul este un
                                                    'număr marcator' atunci forma
                                                    ascunsă a formei punctului
                                                    flotant nu trebuie tipărită.
                INC   HL                            Se incrementează indicatorul
                                                    pentru următoarea trecere.
                CP    +0D                           Este caracterul un 'carriage
                                                    return'?
                JR    Z,18B4,OUT-LINE6              Salt dacă este.
                EX    DE,HL                         Schimbare indicator în DE.
                CALL  1937,OUT-CHAR                 Se tipăreste caracterul.
                JR    1894,OUT-LINE4                Ciclare buclă pînă la sfîrsit
                                                    cu cîte o nouă trecere.
```

Acum linia a fost tipărită.

```
18B4 OUT-LINE6  POP   DE                            Se reface registrul pereche
                RET                                 DE si revenire.
```

THE 'NUMBER' SUBROUTINE (SUBRUTINA 'NUMAR')
Dacă registrul A contine 'număr marcator' atunci registrul pereche HL este avansat peste forma punctului flotant.

```
18B6 NUMBER     CP    +0E                           Este caracterul un 'număr
                RET   NZ                            marcator'? Dacă nu - revenire
                INC   HL                            Se avansează indicatorul de
                INC   HL                            sase ori astfel încît să
                INC   HL                            treacă peste 'numărul
                INC   HL                            marcator' si cele cinci
                INC   HL                            locatii să conțină forma
                INC   HL                            punctului flotant.
                LD    A,(HL)                        Se aduce codul curent înainte
                RET                                 de revenire.
```

THE 'PRINT A FEATCHING CHARACTER' SUBROUTINE (SUBRUTINA 'TIPARIREA UNUI CARACTER PILPIITOR')
'Eroarea cursor' si 'modul cursor' se tipăresc folosind această subrutină.

```
18C1 OUT-FLASH  EXX                                 Salvare registrii curenti.
                LD    HL,(ATTR-T)                   Se salvează ATTR-T & MASK-T
                PUSH  HL                            în stiva calculatorului.
                RES   7,H                           Se asigură că FLASH este
                SET   7,L                           activ.
                LD    (ATTR-T),HL                   Se folosesc aceste valori
                                                    modificate ale lui ATTR-T &
                                                    MASK-T.
                LD    HL,+5C91                      Acesta este P-FLAG.
                LD    D,(HL)                        Si P-FLAG se salvează în
                PUSH  DE                            stiva calculatorului.
                LD    (HL),+00                      Se asigură că este INVERSE 0,
                                                    OVER 0 si nu PAPER 9, nici
                                                    INK 9.
                CALL  09F4,PRINT-OUT                Caracterul este tipărit.
                POP   HL                            Se reface valoarea anterioară
                LD    (P-FLAG),H                    a lui P-FLAG.
                POP   HL                            De asemenea se refac vechile
                LD    (ATTR-T),HL                   valorie ale lui ATTR-T &
                EXX                                 MASK-T înainte de revenire.
                RET
```

THE 'PRINT THE CURSOR' SUBROUTINE (SUBRUTINA 'TIPARIRE CURSOR')
Se face o revenire dacă nu este locul potrivit pentru tipărirea cursorului, dar dacă este, atunci se va tipări unul dintre 'C', 'E', 'G', 'K' sau 'L'.

```
18E1 OUT-CURS   LD    HL,(K-CUR)                    Se aduce adresa cursorului
                AND   A                             dar se execută revenire dacă
                SBC   HL,DE                         nu s-a considerat locul
                RET   NZ                            corect.
                LD    A,(MODE)                      Se aduce valoarea curentă a
```

```
                RLC   A                    lui MODE si se dubleazà.
                JR    Z,18F3,OUT-C-1       Satlt înainte dacà nu se
                ADD   A,+43                lucreazà cu modul Extins sau
                                           Grafic.
                JR    1909,OUT-C-2         Salt înainte pentru a-l
                                           tipàri.
18F3 OUT-C-1    LD    HL,+5C3B             Acesta este FLAGS.
                RES   3,(HL)               Semnal 'mod K'.
                LD    A,+4B                Caracterul 'K'.
                BIT   2,(HL)               Salt înainte pentru a tipàri
                JR    Z,1909,OUT-C-2       'K' dacà 'tipàrirea se va
                                           face în mod K'.
                SET   3,(HL)               'Tipàrirea se va face în mod
                                           L'; asa cà semnalizare 'mod
                                           L'.
                INC   A                    Formare caracter 'L'.
                BIT   3,(FLAGS2)           Salt înainte dacà nu este în
                JR    Z,1909,OUT-C-2       'mod C'.
                LD    A,+43                Caracterul 'C'.
1909 OUT-C-2    PUSH  DE                   Se salveazà registrul pereche
                CALL  18C1,OUT-FLASH       DE pînà cînd cursorul este
                POP   DE                   tipgrit - pîlpîind.
                RET                        Revenire odatà ce s-a fàcut.
```

Notà: Aceasta este actiunea de considerare a cursorului - literà de tipàrit care determinà modul 'K' sau 'L/C'.

THE 'LN-FETCH' SUBROUTINE (SUBRUTINA 'LN-FETCH')
Aceastà subrutinà este introdusà cu registrul pereche HL adresînd o variabilà sistem - S-TOP sau E-PPC.
Subrutina revine cu variabila sistem continînd numàrul de linie al liniei urmàtoare.

```
190F LN-FETCH   LD    E,(HL)               Se colectează numàrul de
                INC   HL                   linie continut de variabila
                                           sistem.
                LD    D,(HL)
                PUSH  HL                   Indicatorul este salvat.
                EX    DE,HL                Numàrul de liniei este mutat
                INC   HL                   în registrul pereche HL si
                                           incrementat.
                CALL  196E,LINE-ADDR       S-a gàsit adresa de început a
                                           acestei linii, sau a liniei
                                           urmàtoare, dacà actualul
                                           numàr de linie nu a fost
                                           folosit.
                CALL  1695,LINE-NO         Acel numàr de linie este adus
                POP   HL                   Se reface indicatorul
                                           variabilei sistem.
```

Punctul de intrare LN-STORE este folosit de EDITOR.

```
191C LN-STORE   BIT   5,(FLAGX)            Revenire dacà este 'mod
                RET   NZ                   INPUT'; altfel se procedează
                LD    (HL),B               la introducerea numàrului de
                DEC   HL                   linie în două locatii ale
                LD    (HL),E               variabilei sistem.
                RET                        Revenire.
```

THE 'PRINTING CHARACTERS IN A BASIC LINE' SUBROUTINE (SUBRUTINA 'TIPARIRE CARACTERE INTR-O LINIE BASIC')
Toate codurile de caractere/semne dintr-o linie BASIC sînt tipàrite prin apelàri repetate ale acestei subrutine.
Punctul de intrare OUT-SP-NO este utilizat cînd se tipàresc numere de linii care cer spatii de fond.

```
1925 OUT-SP-2   LD    A,E                  Registrul A va contine +20
                                           pentru un spatiu si +FF
                                           pentru nici un spatiu.
                AND   A                    Se testează valoarea si se
                RET   M                    revine dacà nu este spatiu.
                JR    1937,OUT-CHAR        Salt înainte pentru tipàrire
                                           spatiu.
192A OUT-SP-NO  XOR   A                    Se sterge registrul A.
```

Registrul pereche HL contine numàrul liniei si registrul BC contine valoarea pentru 'scàdere repetatà'. (BC contine '-1000, -100 sau -10',)

```
192B OUT-SP-1   ADD   HL,BC                'Incercare scàdere'.
```

```
                INC   A                 Se numără fiecare 'încercare'
                JR    C,192B,OUT-SP-1   Salt înapoi pînă la epuizare.
                SBC   HL,BC             Se reface ultima 'scădere' si
                DEC   A                 se decrementează.
                JR    Z,1925,OUT-SP-2   Dacă nu este posibilă nici o
                                        'scădere', salt înapoi pentru
                                        a vedea dacă nu trebuie
                                        tipărit un spatiu.
                JP    15EF,OUT-CODE     In caz contrar, tipărire
                                        digit.
```

Punctul de intrare OUT-CHAR este folosit pentru toate caracterele, semnele si caracterele de control.

```
1937 OUT-CHAR   CALL  2DD3,NUMERIC      Readucerea transportului
                                        resetat dacă se tratează cod
                                        digit.
                JR    NC,196C,OUT-CH-3  Salt înainte pentru tipărire
                                        digit.
                CP    +21               De asemenea se tipăresc
                JR    C,196C,OUT-CH-3   caracterele de control si
                                        'spatiu'.
                RES   2,(FLAGS)         Semnal 'tiărire în mod K'.
                CP    +CB               Salt înainte dacă se lucrează
                JR    Z,196C,OUT-CH-3   cu simbolul 'THEN'.
                CP    +3A               Salt înainte dacă nu se
                JR    NZ,195A,OUT-CH-1  lucrează cu ':'.
                BIT   5,(FLAGX)         Salt înainte pentru a tipări
                JR    NZ,1968,OUT-CH-2  ':' dacă este 'mod INPUT'.
                BIT   2,(FLAGS2)        Salt înainte dacă ':' 'nu
                JR    Z,196C,OUT-CH-3   este in limite', adică un
                                        marcator inter-instructiuni.
                JR    1968,OUT-CH-2     Semnul ':' este între limite
                                        si acum poate fi tipărit.
195A OUT-CH-1   CP    +22               Se acceptă la tipărire toate
                JR    NZ,1968,OUT-CH-2  caracterele, mai putin '"'.
                PUSH  AF                Se salvează codul
                                        caracterului pînă cînd se
                                        schimbă 'mod limite'.
                LD    A,(FLAGS2)        Se aduce FLAG2 si se schimbă
                XOR   +04               bitul 2.
                LD    (FLAGS2),A        Se introduce valoarea
                POP   AF                modificată si se reface codul
                                        caracterului.
1968 OUT-CH-2   SET   2,(FLAGS)         Semnal 'următorul caracter
                                        trebuie tipărit în mod L'.
196D OUT-CH-3   RST   0010,PRINT-A-1    Prezentul caracter este
                RET                     tipărit înainte de revenire.
```

Notă: Este o consecintă a testului asupra caracterului prezent care determină dacă următorul caracter 'se va tipări în mod 'K' sau 'L''.
De asemenea, de notat cum programul nu furnizează ':' în instructiunea REM.

THE 'LINE-ADDR' SUBROUTINE (SUBRUTINA 'LINE-ADDR')
Pentru a da un număr de linie, în registrul pereche HL, această subrutină returnează începutul adresei acelei linii sau 'prima linie de după' si începutului liniei anterioare în registrul pereche DE.
Fanionul zero este setat dacă numărul liniei a fost folosit. Oricum, dacă 'prima linie de după' este substituită atunci fanionul zero este returnat resetat.

```
196E LINE-ADDR  PUSH  HL                Se salvează numărul liniei
                                        dat.
                LD    HL,(PROG)         Se aduce variabila sistem
                LD    D,H               PROG si se transferă adresa
                LD    E,L               în registrul pereche DE.
```

Acum se introduce o buclă pentru a testa numărul liniei al fiecărei linii a programului cu numărul liniei date pînă cînd numărul liniei este egal sau depăsit.

```
1974 LINE-AD-1  POP   BC                Numărul liniei dat.
                CALL  1980,CP-LINES     Se compară numărul de linie
                RET   NC                dat cu numărul liniei
                                        adresate.
                PUSH  BC                Revenire dacă transportul
                CALL  19B8,NEXT-ONE     este resetat; altfel se
                                        adresează numărul liniei
                                        următoare.
```

```
                EX    DE,HL                    Se schimbă indicatorii si
                JR    1974,LINE-AD-1           salt înapoi pentru a lua în
                                               considerare următoarea linie
                                               a programului.
```

THE 'COMPARE LINE NUMBERS' SUBROUTINE (SUBRUTINA 'COMPARARE NUMERE LINIE')
Numărul de linie dat în registrul pereche BC este comparat cu numărul liniei adresate.

```
1980 CP-LINES   LD    A,(HL)                   Se aduce octetul superior al
                CP    B                        numărului de linie adresat si
                RET   NZ                       se compară. Salt dacă nu
                                               coincid.
                INC   HL                       Apoi se compară octetii
                                               inferiori.
                LD    A,(HL)                   Revenire cu fanionul de
                DEC   HL                       transport setat dacă numărul
                CP    C                        liniei adresate trebui încă
                RET                            să ajungă la numărul liniei
                                               date.
```

THE 'FIND EACH STATEMENT' SUBROUTINE (SUBRUTINA 'GASIREA FIECAREI INSTRUCTII'
Această subrutină are două functii distincte.

i. Poate fi folosită pentru a afla 'D'-a instructiune dintr-o linie BASIC - revenind cu registrul pereche HL adresînd locatia dinainte de începutul instructiei si cu fanionul zero setat.

ii. De asemenea, subrutina se poate folosi pentru a afla o instructiune, dacă există, care începe cu codul unui simbol (în registrul E).

```
                INC   HL                       Nu este utilizat.
                INC   HL
                INC   HL
    EACH-STMT   LD    (CH-ADD),HL              Se setează CH-ADD pe octetul
                                               curent.
                LD    C,+00                    Setare fanion 'limite off'.
```

Se introduce o buclă pentru tratarea fiecărei instructiuni dintr-o linie BASIC.

```
1990 EACH-S-1   DEC   D                        Se decrementează 'D' si se
                RET   Z                        revine dacă instructiunea
                                               cerută nu a fost găsită.
                RST   0020,NEXT-CHAR           Se aduce codul următorului
                CP    E                        caracter si salt dacă nu
                JR    NZ,199A,EACH-S-3         coincide cu codul simbolului
                                               dat.
                AND   A                        Dacă coincide, atunci se
                RET                            revine cu fanioanele zero si
                                               de transport resetate.
```

Acum se introduce o altă buclă pentru a considera fiecare caracter din linie, pentru a determina unde se termină instructiunea.

```
1998 EACH-S-2   INC   HL                       Se măreste indicatorul si se
                LD    A,(HL)                   aduce noul cod.
199A EACH-S-3   CALL  1886,NUMBER              Salt peste orice număr.
                LD    (CH-ADD),HL              Se măreste CH-ADD.
                CP    +22                      Salt înainte în cazul în care
                JR    NZ,19A5,EACH-S-4         caracterul nu este '"'.
                DEC   C                        In caz contrar se setează
                                               'fanionul limite'.
19A5 EACH-S-4   CP    +3A                      Salt înainte în caz în care
                JR    Z,19AD,EACH-S-5          caracterul este ':'-
                CP    +CB                      Salt înainte dacă nu este
                JR    NZ,19B1,EACH-S-6         simbolul 'THEN'.
19AD EACH-S-5   BIT   0,C                      Se citeste 'fanionul cote' si
                JR    Z,1990,EACH-S-1          salt înapoi la sfîrsitul
                                               fiecărei instructii (începînd
                                               după 'THEN').
19B1 EACH-S-6   CP    +0D                      Salt înapoi dacă nu este
                JR    NZ,1998,EACH-S-2         sfîrsitul unei linii BASIC.
                DEC   D                        Se decrementează contorul de
                SCF                            instructiuni si se setează
                RET                            fanionul de transport înainte
                                               de a se reveni.
```

THE 'NEXT-ONE' SUBROUTINE (SUBRUTINA 'URMATOAREA')

Această subrutină se poate folosi pentru a afla 'linia următoare' în spatiul programului sau 'variabila următoare' în spatiul variabilelor. Subrutina este furnizorul pentru sase tipuri diferite de variabile care sînt utilizate în sistemul SPECTRUM.

```
19B8 NEXT-ONE   PUSH  HL              Se salvează adresa liniei
                                      curente sau a variabilei.
                LD    A,(HL)          Se aduce primul octet.
                CP    +40             Salt înainte dacă se caută
                JR    C,19D5,NEXT-0-3 'linia următoare'.
                BIT   5,A             Salt înainte dacă se caută
                JR    Z,19D6,NEXT-0-4 următorul sir.
                ADD   A,A             Salt înainte cu un numeric
                JP    M,19C7,NEXT-0-1 simplu si varibilele
                                      FOR-NEXT.
                CCF                   Numai pentru variabile
                                      numerice cu nume lung.
19C7 NEXT-0-1   LD    BC,+0005        O varibilă numerică va ocupa
                JR    NC,19CE,NEXT-0-2 cinci locatii dar variabila
                LD    C,+12           de control FOR-NEXT va avea
                                      nevoie de nouă locatii.
19CE NEXT-0-2   RLA                   Fanionul de transport va fi
                                      resetat doar pentru
                                      variabilele cu nume lung;
                                      pînă se găseste caracterul
                                      final al numelui lung.
                INC   HL              Se incrementează indicatorul
                LD    A,(HL)          si se aduce noul cod.
                JR    NC,19CE,NEXT-0-2 Salt înapoi dacă codul
                                      anterior nu a fost ultimul
                                      cod al numelui variabilei.
                JR    19DB,NEXT-0-5   Acum salt înainte (BC =
                                      +0005 sau +0012).
19D5 NEXT-0-3   INC   HL              Se trece peste octetul
                                      inferior al numărului liniei.
19D6 NEXT-0-4   INC   HL              Acum se indică octetul
                                      inferior al lungimii.
                LD    C,(HL)          Se aduce lungimea în
                INC   HL              registrul pereche BC.
                LD    B,(HL)
                INC   HL              Se dă permisiunea octetului
                                      cuprinzător.
```

In toate cazurile s-a găsit adresa liniei 'următoare' sau a variabilei.

```
19DB NEXT-0-5   ADD   HL,BC           Se indică primul octet al
                                      liniei 'următoare' sau al
                                      variabilei.
                POP   DE              Se aduce adresa celei
                                      anterioare si se continuă î
                                      subrutina de 'diferentiere'.
```

THE 'DIFFERENCE' SUBROUTINE (SUBRUTINA 'DIFERENTIERE')
'Lungimea' dintre două 'începuturi' este formată în registrul pereche BC. Indicatorii sînt refăcuti dar ei sînt returnati schimbati.

```
19DD DIFFER     AND   A               Pregătire pentru scădere.
                SBC   HL,DE           Se gaseste lungimea de la un
                LD    B,H             'început' la următorul si ea
                LD    C,L             se trece în registrul pereche
                                      BC.
                ADD   HL,DE           Se reformează adresa si se
                EX    DE,HL           schimbă înainte de revenire.
                RET
```

THE 'RECLAIMING' SUBROUTINE (SUBRUTINA 'REFACERE')
Punctul de intrare RECLAIM-1 este utilizat cînd adresa primei locatii ce trebuie refăcută este în registrul pereche DE si adresa primei locatii lăsată izolată este în registrul pereche HL. Punctul de intrare RECLAIM-2 este utilizat cînd registrul pereche indică prima locatie ce trebuie refăcută si registrul pereche BC contine numărul octetilor care trebuie refăcuti.

```
19E5 RECLAIM-1  CALL  19DD,DIFFER     Se foloseste rutina de
                                      'diferentiere' pentru a
                                      realiza valoarea
                                      corespunzătoare.
19E8 RECLAIM-2  PUSH  BC              Se salvează numărul octetilor
                                      care trebuie refăcuti.
                LD    A,B             Indicatorii tuturor
```

```
                CPL                     variabilelor sistem deasupra
                LD    B,A               zonei trebuie redusi cu 'BC'
                LD    A,C               asa cà acest numàr este
                CPL                     complementat de 2 ori înainte
                LD    C,A               ca indicatorii sà fie
                INC   BC                schimbati.
                CALL  1664,POINTERS
                EX    DE,HL             Adresa 'primei locatii'
                POP   HL                revine în registrul pereche
                ADD   HL,DE             DE si se reformeazà adresa
                                        primei locatii izolate.
                PUSH  DE                Se salveazà 'prima locatie'
                LDIR                    pînà cînd intervine actuala
                POP   HL                refacere.
                RET                     Acum se revine.
```

THE 'E-LINE-NO' SUBROUTINE (SUBRUTINA 'E-LINE-NO')
Aceastà subrutinà este folositä pentru a citi numàrul de linie al liniei din zona de editare. Dacà nu este nici un numàr de linie, adicà o linie Basic directà, atunci se considerà cà numàrul liniei este zero.
In toate cazurile, numàrul liniei este returnat în registrul pereche BC.

```
19FB E-LINE-NO  LD    HL,(E-LINE)       Punctarea indicatorilor pe
                                        linia de editare.
                DEC   HL                Se seteazà CH-ADD sà indice
                LD    (CH-ADD),HL       locatia dinaintea oricàrui
                                        numàr.
                RST   0020,NEXT-CHAR    Se trece primul cod în
                                        registrul A.
                LD    HL,+5C92          Înainte de a considera codul
                LD    (STKEND),HL       se realizeazà în zona
                                        memoriei calculatorului o
                                        zonà de stivà temporarà.
                CALL  2D3B,INT-TO-FP    Acum se citesc digitii
                                        numàrului liniei. Revenire
                                        zero dacà nu este nici un
                                        numàr.
                CALL  2DA2,FP-TO-BC     Se comprimà numàrul liniei în
                                        registrul pereche BC.
                JR    C,1A15,E-L-1      Salt înainte dacà numàrul
                                        depàseste '65,536'.
                LD    HL,+D8F0          Altfel se testeazà fatà de
                ADD   HL,BC             '10,000'.
                JP    C,1C8A,REPORT-C   Se dà prezentarea C dacà este
                                        peste '9,999'.
                JP    16C5,SET-STK      Revenire prin SET-STK, care
                                        reface stiva calculatorului
                                        în locul cuvenit.
```

THE 'REPORT AND LINE NUMBER PRINTING' SUBROUTINE (SUBRUTINA 'TIPARIRE PREZENTARE SI NUMAR DE LINIE')
Punctul de intrare OUT-NUM-1 va conduce ca numàrul din registrul pereche BC sà fie tipàrit. Orice valoare peste '9,999' nu va fi tipàritä corect.
Punctul de intrare OUT-NUM-2 va conduce ca numàrul indirect adresat de registrul pereche HL sà fie tipàrit. In acest moment orice spatiu de fond este necesar. Si aici numàrul limitä pentru tipàrirea corectà este '9,999'.

```
1A1B OUT-NUM-1  PUSH  DE                Se salveazà registrii
                PUSH  HL                implicati în subrutinà.
                XOR   A                 Se sterge registrul A.
                BIT   7,B               Salt înainte pentru a tipàri
                JR    NZ,1A42,OUT-NUM-4 un zero mai degrabà decît un
                                        '-2' cînd se prezintà într-o
                                        linie de editare.
                LD    H,B               Se mutà numàrul în registrul
                LD    L,C               pereche HL.
                LD    E,+FF             Fanion 'nici un spatiu de
                                        fond'.
                JR    1A30,OUT-NUM-3    Salt înainte pentru a tipàri
                                        numàrul.
1A28 OUT-NUM-2  PUSH  DE                Se salveazà registrul pereche
                                        DE.
                LD    D,(HL)            Se aduce numàrul în registrul
                INC   HL                pereche DE si se salveazà
                LD    E,(HL)            indicatorul (màrit).
                PUSH  HL
                EX    DE,HL             Se mutà numàrul în registrul
                LD    E,+20             pereche HL si se semnaleazà
```

```
                                    'spatiile de  fond trebuie
                                    tipărite'.
1A30 OUT-NUM-3  LD    BC,+FC18      Acesta este '-1.000'.
                CALL  192A,OUT-SP-NO Se tipăreste primul digit.
                LD    BC,+FF9C      Acesta este '-100'.
                CALL  192A,OUT-SP-NO Se tipăreste al doilea digit.
                LD    C,+F6         Acesta este '-10'.
                CALL  192A,OUT-SP-NO Se tipăreste al treilea digit
                LD    A,L           Mutarea oricărei părti rămase
                                    a numărului în registrul A.
1A42 OUT-NUM-4  CALL  15EF,OUT-CODE Se tipăreste digitul.
                POP   HL            Se refac registrii înainte de
                POP   DE            revenire.
                RET
```

BASIC LINE AND COMMAND INTERPRETATION (INTERPRETAREA LINIEI BASIC SI A COMENZILOR)

THE SYNTAX TABLES (TABELELE DE SINTAXA)

i. The offset table (tabelul de deplasament)
Pentru fiecare 50 de comenzi BASIC există o valoare a deplasamentului.

			comanda	adresa				comanda	adresa
1A48	DEFB	+B1	DEF FN	1AF9	1A61	DEFB	+94	BORDER	1AF5
1A49	DEFB	+CB	CAT	1B14	1A62	DEFB	+56	CONTINUE	1AB8
1A4A	DEFB	+BC	FORMAT	1B06	1A63	DEFB	+3F	DIM	1AA2
1A4B	DEFB	+BF	MOVE	1B0A	1A64	DEFB	+41	REM	1AA5
1A4C	DEFB	+C4	ERASE	1B10	1A65	DEFB	+2B	FOR	1A90
1A4D	DEFB	+AF	OPEN #	1AFC	1A66	DEFB	+17	GO TO	1A7D
1A4E	DEFB	+B4	CLOSE #	1B02	1A67	DEFB	+1F	GO SUB	1A86
1A4F	DEFB	+93	MERGE	1AE2	1A68	DEFB	+37	INPUT	1A9F
1A50	DEFB	+91	VERIFY	1AE1	1A69	DEFB	+77	LOAD	1AE0
1A51	DEFB	+92	BEEP	1AE3	1A6A	DEFB	+44	LIST	1AAE
1A52	DEFB	+95	CIRCLE	1AE7	1A6B	DEFB	+0F	LET	1A7A
1A53	DEFB	+98	INK	1AE8	1A6C	DEFB	+59	PAUSE	1AC5
1A54	DEFB	+98	PAPER	1AEC	1A6D	DEFB	+2B	NEXT	1A98
1A55	DEFB	+98	FLASH	1AED	1A6E	DEFB	+43	POKE	1AB1
1A56	DEFB	+98	BRIGHT	1AEE	1A6F	DEFB	+2D	PRINT	1A9C
1A57	DEFB	+98	INVERSE	1AEF	1A70	DEFB	+51	PLOT	1AC1
1A58	DEFB	+98	OVER	1AF0	1A71	DEFB	+3A	RUN	1AAB
1A59	DEFB	+98	OUT	1AF1	1A72	DEFB	+6D	SAVE	1ADF
1A5A	DEFB	+7F	LPRINT	1AD9	1A73	DEFB	+42	RANDOMIZE	1AB5
1A5B	DEFB	+81	LLIST	1ADC	1A74	DEFB	+0D	IF	1A81
1A5C	DEFB	+2E	STOP	1A8A	1A75	DEFB	+49	CLS	1ABE
1A5D	DEFB	+6C	READ	1AC9	1A76	DEFB	+5C	DRAW	1AD2
1A5E	DEFB	+6E	DATA	1ACC	1A77	DEFB	+44	CLEAR	1ABB
1A5F	DEFB	+70	RESTORE	1ACF	1A78	DEFB	+15	RETURN	1A8D
1A60	DEFB	+48	NEW	1AA8	1A79	DEFB	+5D	COPY	1AD6

ii. The parameter table (Tabelul parametru)
Pentru fiecare 50 de comenzi BASIC există pînă la opt intrări în tabelul parametru. Aceste intrări cuprind detaliile clasei de comenzi, solicită separatori si adresele rutinelor comenzilor.

1A7A	P-LET	DEFB +01	CLASS-01
		DEFB +3D	'='
		DEFB +02	CLASS-02
1A7D	P-GO-TO	DEFB +06	CLASS-06
		DEFB +00	CLASS-00
		DEFB +67,+1E	GO-TO,1E67
		DEFB +06	CLASS-06
		DEFB +CB	'THEN'
		DEFB +05	CLASS-05
		DEFB +F0,+1C	IF,1CF0
1A86	P-GO-SUB	DEFB +06	CLASS-06
		DEFB +00	CLASS-00
		DEFB +ED,+1E	GO-SUB,1EED
1A8A	P-STOP	DEFB +00	CLASS-00
		DEFB +EE,+1C	STOP,1CEE
1A8D	P-RETURN	DEFB +00	CLASS-0
		DEFB +23,+1F	RETURN,1F23
		DEFB +04	CLASS-04
		DEFB +3D	'='
		DEFB 06	CLASS-06
		DEFB +CC	'TO'
		DEFB +06	CLASS-06
		DEFB +05	CLASS-05
		DEFB +03,+1D	FOR,1D03
1A98	P-NEXT	DEFB +04	CLASS-04
		DEFB +00	CLASS-00
		DEFB +AB,+1D	NEXT,1DAB
1A9C	P-PRINT	DEFB +05	CLASS+05
		DEFB +CD,+1F	PRINT,1FCD
1A9F	P-INPUT	DEFB +05	CLASS-05
		DEFB +89,+20	INPUT,2089
1AA2	P-DIM	DEFB +05	CLASS-05
		DEFB +02,+2C	DIM,2C02
1AA5	P-REM	DEFB +05	CLASS-05
		DEFB +B2,+1B	REM,1BB2
1AA8	P-NEW	DEFB +00	CLASS-00
		DEFB +B7,+11	NEW,11B7
1AAB	P-RUN	DEFB +03	CLASS-03
		DEFB +A1,+1E	RUN,1EA1

06821

```
1AAE P-LIST       DEFB  +05             CLASS-05
                  DEFB  +F9,+17         LIST,17F9
1AB1 P-POKE       DEFB  +08             CLASS-08
                  DEFB  +00             CLASS-00
                  DEFB  +80,+1E         POKE,1E80
1AB5 P-RANDOM     DEFB  +03             CLASS-03
                  DEFB  +4FF,+1E        RANDOMIZE,1E4F
1AB8 P-CONT       DEFB  +00             CLASS-00
                  DEFB  +5F,+1E         CONTINUE,1E5F
1ABB P-CLEAR      DEFB  +03             CLASS-03
                  DEFB  +AC,+1E         CLEAR,1EAC
1ABE P-CLS        DEFB  +00             CLASS-00
                  DEFB  +6B,+0D         CLS,0D6B
1AC1 P-PLOT       DEFB  +09             CLASS-09
                  DEFB  +00             CLASS-00
                  DEFB  +DC,+22         PLOT,22DC
1AC5 P-PAUSE      DEFB  +06             CLASS-06
                  DEFB  +00             CLASS-00
                  DEFB  +3A,+1F         PAUSE,1F3A
1AC8 P-READ       DEFB  +05             CLASS-05
                  DEFB  +ED,+1D         READ,1DED
1ACC P-DATA       DEFB  +05             CLASS-05
                  DEFB  +27,+1E         DATA,1E27
1ACF P-RESTORE    DFB   +03             CLASS-03
                  DEFB  +42,+1E         RESTORE,1E42
1AD2 P-DRAW       DEFB  +09             CLASS-09
                  DEFB  +05             CLASS-05
                  DEFB  +82,+23         DRAW,2382
1AD6 P-COPY       DEFB  +00             CLASS-00
                  DEFB  +AC,+0E         COPY,0EAC
1AD9 P-LPRINT     DEFB  +05             CLASS-05
                  DEFB  +C9,+1F         LPRINT,1FC9
1ADC P-LIST       DEFB  +05             CLASS-05
                  DEFB  +F5,+17         LLIST,17F5
1ADF P-SAVE       DEFB  +0B             CLASS-0B
1AE0 P-LOAD       DEFB  +0B             CLASS-0B
1AE1 P-VERIFY     DEFB  +0B             CLASS-0B
1AE2 P-MERGE      DEFB  +0B             CLASS-0B
1AE3 P-BEEP       DEFB  +08             CLASS-08
                  DEFB  +00             CLASS-00
                  DEFB  +F8,+03         BEEP,03F8
1AE7 P-CIRCLE     DEFB  +09             CLASS-09
                  DEFB  +05             CLASS-05
                  DEFB  +20,+23         CIRCLE,2320
1AEB P-INK        DEFB  +07             CLASS-07
1AEC P-PAPER      DEFB  +07             CLASS-07
1AED P-FLASH      DEFB  +07             CLASS-07
1AEE P-BRIGHT     DEFB  +07             CLASS-07
1AEF P-INVERSE    DEFB  +07             CLASS-07
1AF0 P-OVER       DEFB  +07             CLASS-07
1AF1 P-OUT        DEFB  +08             CLASS-08
                  DEFB  +00             CLASS-00
                  DEFB  +7A,+1E         OUT,1E7A
1AF5 P-BORDER     DEFB  +06             CLASS-06
                  DEFB  +00             CLASS-00
                  DEFB  +94,+22         BORDER,2294
1AF9 P-DEF-FN     DEFB  +05             CLASS-05
                  DEFB  +60,+1F         DEF-FN,1F60
1AFC P-OPEN       DEFB  +06             CLASS-06
                  DEFB  +2C             ,
                  DEFB  +0A             CLASS-0A
                  DEFB  +00             CLASS-00
                  DEFB  +36,+17         OPEN,1736
1B02 P-CLOSE      DEFB  +06             CLASS-06
                  DEFB  +00             CLASS-00
                  DEFB  +E5,+16         CLOSE,16E5
1B06 P-FORMAT     DEFB  +0A             CLASS-0A
                  DEFB  +00             CLASS-00
                  DEFB  +93,+17         CAT-ETC,1793
1B0A P-MOVE       DEFB  +0A             CLASS-0A
                  DEFB  +2C             ,
                  DEFB  +0A             CLASS-0A
                  DEFB  +00             CLASS-00
                  DEFB  +93,+17         CAT-ETC,1793
1B10 P-ERASE      DEFB  +0A             CLASS-0A
                  DEFB  +00             CLASS-00
                  DEFB  +93,+17         CAT-ETC,1793
1B14 P-CAT        DEFB  +00             CLASS-00
                  DEFB  +93,+17         CAT-ETC,1793
```

Notă: Cererile pentru diferitele categorii de comenzi sînt următoarele:

CLASS-00 - Nu mai sînt operatori în continuare.
CLASS-01 - Se foloseste în LET. Cerere de variabilă.
CLASS-02 - Se foloseste în LET. Trebuie să urmeze o expresie numerică sau un sir.
CLASS-03 - Trebuie să urmeze o expresie numerică. In caz de eroare se foloseste zero.
CLASS-04 - Trebuie să urmeze un singur caracter variabil.
CLASS-05 - Trebuie dat un set de numere.
CLASS-06 - Trebuie să urmeze o expresie numerică.
CLASS-07 - Se tratează numerele de culori.
CLASS-08 - Trebuie să urmeze două expresii numerice, separate prin virgulă.
CLASS-09 - In privinta CLASS-08, doar numerele de culori pot preceda expresii.
CLASS-0A - Trebuie să urmeze o expresie sir.
CLASS-0B - Se tratează rutinele legate de casetă.

THE 'MAIN PARSER' OF THE BASIC INTERPRETER ('ANALIZA PRINCIPALA' A INTERPRETORULUI BASIC)
Rutina de analizare a interpretorului BASIC intră pe la LINE-SCAN cînd sintaxa a fost verificată, si pe la LINE-RUN cînd se execută un program BASIC care are una sau mai multe instructii.
Se consideră pe rînd fiecare instructiune si se foloseste variabila sistem CH-ADD pentru a indica fiecare cod al instructiunii ca si cum s-ar afla în spatiul programului sau în spatiul editat.

```
1B17 LINE-SCAN   RES   7,(FLAGS)          Semnal 'verificare sintaxă'.
                 CALL  19FB,E-LINE-NO     CH-ADD va indica primul cod
                                          după oricare număr de linie.
                 XOR   A                  Variabila sistem SUBPPC se
                 LD    (SUBPPC),A         initializează cu +00 si
                 DEC   A                  ERR-NR cu +FF.
                 LD    (ERR-NR),A
                 JR    1B29,STMT-L-1      Salt înainte pentru a lua în
                                          considerare prima
                                          instructiune a liniei.
```

THE STATEMENT LOOP (INSTRCUTIUNEA DE CICLARE)
Se consideră pe rînd fiecare instructiune pînă cînd se atinge sfîrsitul liniei.

```
1B28 STMT-LOOP   RST   0020,NEXT-CHAR     Avans CH-ADD de-a lungul
                                          liniei.
1B29 STMT-L-1    CALL  16BF,SET-WORK      Spatiul de lucru este sters.
                 INC   (SUBPPC)           Se incrementează SUBPPC la
                                          fiecare trecere prin buclă.
                 JP    M,1C8A,REPORT-C    Sînt permise doar '127' de
                                          instructiuni într-o singură
                                          linie.
                 RST   0018,GET-CHAR      Se aduce un caracter.
                 LD    B,+00              Se sterge registrul pentru
                                          mai tîrziu.
                 CP    +0D                Salt în cazul în care
                 JR    Z,1B83,LINE-END    caracterul este un 'carriage
                                          return'.
                 CP    +3A                Reîntoarcere în buclă dacă
                 JR    Z,1B28,STMT-LOOP   este ':'.
```

A fost identificată o instructiune, asa că, prima dată se consideră comanda sa initială.

```
                 LD    HL,+1B76           Se încarcă stiva masinii cu
                 PUSH  HL                 adresa de revenire - STMT-RET
                 LD    C,A                Se salvează temporar comanda
                 RST   0020,NEXT-CHAR     în registrul C pînă CH-ADD a
                 LD    A,C                avansat iar.
                 SUB   +CE                Se reduce codul comenzii cu
                                          +CE; se dă ordinul +00 pînă
                                          la +31 pentru 50 de comenzi.
                 JP    C,1BCA,REPORT-C    Se comunică eroarea
                                          corespunzătoare, dacă nu este
                                          un cod de comandă.
                 LD    C,A                Se mută codul comenzii în
                                          registrul pereche BC (B
                                          contine +00).
                 LD    HL,+1A48           Adresa de bază a tabelului
```

```
                        ADD   HL,BC          sintaxei deplasamentului.
                        LD    C,(HL)         Deplasamentul cerut este
                        ADD   HL,BC          trecut în registrul C si se
                                             foloseste la calculul adresei
                                             de bază pentru intrările
                                             comenzii în tabelul
                                             parametru.
                        JR    1B55,GET-PARAM Salt înainte cu această
                                             adresă în bucla de scanare.
```

Fiecare din rutinele categoriei de comenzi aplicabile prezentei comenzi se execută pe rînd. De asemenea, se consideră orice cerere de separatori.

```
1B52 SCAN-LOOP          LD    HL,(T-ADDR)    Se încarcă în tabelul
                                             parametru indicatorul
                                             temporar al intrărilor.
1B55 GET-PARAM          LD    A,(HL)         Se încarcă pe rînd fiecare
                                             intrare.
                        INC   HL             Se incrementează indicatorul
                        LD    (T-ADDR),HL    intrărilor pentru următorul
                                             pas.
                        LD    BC,+1B52       Se încarcă stiva masinii cu
                        PUSH  BC             adresa de revenire SCAN-LOOP.
                        LD    C,A            Se copiază intrarea în
                                             registrul C pentru mai
                                             tîrziu.
                        CP    +20            Salt înainte dacă intrarea
                        JR    NC,1B6F,SEPARATOR este un 'separator'.
                        LD    HL,+1C01       Se încarcă adresa de bază a
                                             tabelului 'clasei de
                                             comenzi'.
                        LD    B,+00          Se sterge registrul B si se
                        ADD   HL,BC          indexează în tabel.
                        LD    C,(HL)         Se aduce deplasamentul si se
                        ADD   HL,BC          calculează adresa de început
                                             a rutinei clasei de comenzi
                                             cerute.
                        PUSH  HL             Se încarcă această adresă în
                                             stiva calculatorului.
                        RST   0018,GET-CHAR  Înainte de a executa un salt
                        DEC   B              indirect la rutina clasei
                        RET                  comenzii, se trece codul
                                             comenzii în registrul A iar
                                             registrul B se încarcă cu
                                             +FF.
```

THE 'SEPARATOR' SUBROUTINE (SUBRUTINA 'SEPARATOR')
Prezentarea - 'Nonsens în BASIC' este dată dacă nu există cererea de separator. De notat că atunci cînd sintaxa a fost verificată, această prezentare nu apare pe ecran - doar 'error marker' (marcator de eroare).

```
1B6F SEPARATOR          RST   0018,GET-CHAR  Caracterul curent este adus
                        CP    C              si comparat cu intrarea în
                                             tabelul parametru.
                        JP    NZ,1C8A,REPORT-C Se comunică eroare dacă nu
                                             coincid.
                        RST   0020,NEXT-CHAR Se trece un caracter corect
                        RET                  si revenire.
```

THE 'STMT-RET' SUBROUTINE (SUBRUTINA 'STMT-RET')
După interpretarea corectă a instructiunii se revine la acest punct de intrare.

```
1B76 STMT-RET           CALL  1F54,BREAK-KEY Se apasă tasta BREAK după
                                             fiecare instructiune.
                        JR    C,1B7D,STMT-R-1 Salt înainte dacă nu a fost
                                             apăsată.
```

Prezentarea L - 'BREAK în program'

```
1B7B REPORT-L           RST   0008,ERROR-1   Se apelează rutina de tratare
                        DEFB  +14            eroare.
```

Aici se continuă dacă tasta BREAK nu a fost apăsată.

```
1B7D STMT-R-1           BIT   7,(NSPPC)      Salt înainte dacă nu trebuie
                        JR    NZ,1BF4,STMT-NEXT făcut nici un 'salt'.
                        LD    HL,(NEWPPC)    Se aduce numărul 'liniei noi'
                        BIT   7,H            dacă nu se lucrează cu o
```

```
                JR    Z,1B9E,LINE-NEW        instructiune noua în spatiul
                                             de editare.
```

THE 'LINE-RUN' ENTRY POINT (PUNCTUL DE INTRARE 'EXECUTIE LINIEI')
Acest punct de intrare este utilizat întotdeauna cînd o linie din zona de editare trebuie 'executată'. Intr-un asemenea caz fanionul sintaxă/executie (bitul 7 din FLAGS) trebuie setat.
Punctul de intrare este de asemenea utilizat la testarea sintaxei unei linii din zona de editare care are mai mult de o instructiune (bitul 7 din FLAGS va fi resetat).

```
1B8A LINE-RUN   LD    HL,+FFFE               O linie din zona de editare
                LD    (PPC),HL               este considerată ca linia
                                             '-2'.
                LD    HL,(WORKSP)            HL va indica marcatorul de
                DEC   HL                     sfîrsit al zonei de editare
                LD    DE,(E-LINE)            si DE locatia dinainte de
                DEC   DE                     începutul acestei zone.
                LD    A,(NSPPC)              Se aduce numărul următoarei
                JR    1BD1,NEXT-LINE         instructiuni pentru a se
                                             trata înainte de a sări
                                             înainte.
```

THE 'LINE-NEW' SUBROUTINE (SUBRUTINA 'LINIE NOUA')
In program s-a făcut un salt si trebuie găsită adresa de început a noii linii.

```
1B9E LINE-NEW   CALL  196E,LINE-ADDR         S-a găsit adresa de început a
                                             liniei sau a 'primei linii de
                                             după'.
                LD    A,(NSPPC)              Se adună numărul instructiei.
                JR    Z,1BBF,LINE-USE        Salt înainte dacă s-a găsit
                AND   A                      linia cerută; în caz contrar,
                JR    NZ,1BEC,REPORT-N       se verifică validitatea
                                             numărului instructiunii -
                                             trebuie să fie zero.
                LD    B,A                    Se mai verifică dacă 'prima
                LD    A,(HL)                 linie de după' nu este după
                AND   +C0                    actualul 'sfîrsit de program'
                LD    A,B
                JR    Z,1BBF,LINE-USE        Salt înainte cu adresele
                                             valide; altfel se
                                             semnalizează eroare 'OK'.
```

Prezentarea 0 - 'OK'

```
1BB0 REPORT-0   RST   0008,ERROR-1           Se foloseste rutina de
                DEFB  +FF                    tratare eroare.
```

Notă: Evident că nu este o eroare în adevăratul sens - este mai degrabă un salt peste program.

THE 'REM' COMMAND ROUTINE (RUTINA DE COMANDA 'REM')
Adresa de revenire la STMT-RET este coborîtă , avînd ca efect fortarea ignorării restului liniei.

```
1BB2 REM        POP   BC                     Coborîre adresă - STMT-RET.
```

THE 'LINE-END' ROUTINE (RUTINA 'SFIRSIT LINIEI')
Dacă se verifică sintaxa se face o simplă revenire, dar cînd 'se execută', adresa continută de NXTLIN trebuie verificată înainte de a putea fi utilizată.

```
1BB3 LINE-END   CALL  2530,SYNTAX-Z          Revenire dacă s-a verificat
                RET   Z                      sintaxa; dacă nu, se aduce
                LD    HL,(NXTLIN)            adresa în NXTLIN.
                LD    A,+C0                  revenire si dacă adresa este
                AND   (HL)                   după sfîrsitul programului -
                RET   NZ                     'executia' este terminată.
                XOR   A                      Semnalizare 'instructiune
                                             zero' înainte de utilizare.
```

THE 'LINE-USE' ROUTINE (RUTINA 'UTILIZARE LINIEI'.
Această rutină are trei functii: i. Schimbă instructia zero cu instructia '1'; ii. Găseste numărul liniei noi si-l introduce în PPC; & iii. Formează adresa de început a liniei următoare.

```
1BBF LINE-USE   CP    +01                    Instructiunea zero devine
                ADC   A,+00                  instrcutiunea '1'.
                LD    D,(HL)                 Numărul de linie al liniei ce
                INC   HL                     trebuie folosită este adunat
```

```
                LD    E,(HL)          si trecut în PPC.
                LD    (PPC),DE
                INC   HL              Acum se determină 'lungimea'
                LD    E,(HL)          liniei.
                INC   HL
                LD    D,(HL)
                EX    DE,HL           Interschimbarea valorilor.
                ADD   HL,DE           Se formează adresa de început
                INC   HL              a liniei următoare în HL si
                                      în DE - locatia dinainte de
                                      primul caracter al liniei
                                      'următoare'.
```

THE 'NEXT-LINE' ROUTINE (RUTINA 'URMATOAREA LINIE')
La intrare registrul pereche HL indică locatia după sfîrsitul liniei 'următoare' ce va fi tratată, iar registrul pereche DE indică locatia dinaintea primului caracter al liniei. Aceasta se aplică liniilor din zona programului si de asemenea unei linii din spatiul de editare - unde linia următoare va fi aceeasi linie din nou pînă cînd mai sînt instructiuni de interpretat.

```
1BD1 NEXT-LINE  LD    (NXTLIN),HL     Setare NXTLIN pentru a mai
                                      utiliza o dată linia curentă
                                      ce a fost completată.
                EX    DE,HL           Ca de obicei CH-ADD indică
                LD    (CH-ADD),HL     locatia dinaintea primului
                                      caracter considerat.
                LD    D,A             Se aduce numărul instructiei.
                LD    E,+00           Se sterge registrul E dacă se
                                      foloseste EACH-STMT.
                LD    (NSPPC),+FF     Semnalizare 'nu este salt'.
                DEC   D               Numărul instructiei minus unu
                LD    (SUBPPC),D      trece în SUBPPC.
                JP    Z,1B28,STMT-LOOP Acum se poate considera prima
                                      instructiune.
                INC   D               Oricum, trebuie găsită
                CALL  198B,EACH-STMT  adresa de început' pentru
                                      următoarele instructiuni.
                JR    Z,1BF4,STMT-NEXT Salt înainte dacă nu mai
                                      există instructiune.
```

Prezentarea N - 'Instructiune pierdută'

```
1BEC REPORT-N   RST   0008,ERROR-1    Se apelează rutina de tratare
                DEFB  +16             eroare.
```

THE 'CHECK-END' SUBROUTINE (SUBRUTINA 'VERIFICARE SFIRSIT')
Aceasta este o rutină importantă si este apelată din mai multe locuri în programul monitor cînd este verificată sintaxa liniei editate. Scopul rutinei este să dea prezentarea erorii dacă nu s-a completat sfîrsitul instructiunii si pentru a muta în instructiunea următoare dacă sintaxa este corectă.

```
1BEE CHECK-END  CALL  2530,SYNTAX-Z   Nu se porneste dacă nu este
                RET   NZ              verificată sintaxa.
                POP   BC              Se scot adresele pentru
                POP   BC              SCAN-LOOP & STMT-RET înainte
                                      de a continua în STMT-NEXT.
```

THE 'STMT-NEXT' ROUTINE (RUTINA 'STMT-NEXT')
In cazul în care caracterul prezent este un 'carriage return', atunci 'instructiunea următoare' este în 'linia următoare'; dacă ':' este în aceeasi linie si totusi se găseste orice alt caracter atunci este eroare de sintaxă.

```
1BF4 STMT-NEXT  RST   0018,GET-CHAR   Se aduce caracterul prezent.
                CP    +0D             Se consideră 'linia
                JR    Z,1BB3,LINE-END următoare' dacă este un
                                      'carriage return'.
                CP    +3A             Se consideră 'instructiunea
                JP    Z,1B28,STMT-LOOP următoare' dacă este un ':'.
                JP    1C8A,REPORT-C   Altfel a fost o eroare de
                                      sintaxă.
```

THE 'COMMAND CLASS' TABLE (TABELUL 'CLASA DE COMANDA')

I	II	III	I	II	III
1C01	0F	CLASS-00,1C10	1C07	7B	CLASS-06,1C82
1C02	1D	CLASS-01,1C1F	1C08	8E	CLASS-07,1C96

```
1C03   4B     CLASS*02,1C4E        1C09   71     CLASS-08,1C7A
1C04   09     CLASS-03,1C0D        1C0A   B4     CLASS-09,1CBE
1C05   67     CLASS-04,1C6C        1C0B   81     CLASS-0A,1C8C
1C06   0B     CLASS-05,1C11        1C0C   CF     CLASS-0B,1CDB
```

Notă: I - adresa
II - deplasamentul
III - numărul clasei

THE 'COMMAND CLASSES - 00, 03 & 05' ('CLASELE DE COMANDA - 00, 03 & 05')
Comenzile clasei 03 pot fi sau nu urmate de un număr, de exemplu RUN & RUN 200.

```
1C0D CLASS-03      CALL  1CDE,FETCH-NUM        Se aduce un număr, dar dacă
                                               lipseste se foloseste zero.
```

Comenzile clasei 00 nu trebuie să aibă nici un operand, de exemplu COPY & CONTINUE.

```
1C10 CLASS-00      CP    A                     Setare fanion zero pentru mai
                                               tîrziu.
```

Comenzile din clasa 05 pot fi urmate de un set de numere, de exemplu PRINT & PRINT "222".

```
1C11 CLASS-05      POP   BC                    In toate cazurile se scoate
                                               adresa - SCANN-LOOP.
                   CALL  Z,1BEE,CHECK-END      Dacă se tratează comenzi din
                                               clasele 00 & 03 si sintaxa a
                                               fost verificată se trece la
                                               considerarea următoarei
                                               instructiuni.
                   EX    DE,HL                 Se salvează indicatorul de
                                               linie în registrul pereche
                                               DE.
```

THE 'JUMP-C-R' ROUTINE (RUTINA 'JUMP-C-R')
După ce s-au luat în considerare intrările claselor de comandă si intrările separatorului în tabelul parametru se face un salt la rutina de comandă corespunzătoare.

```
1C16 JUMP-C-R      LD    HL,(T-ADDR)           Se aduce indicatorul la
                   LD    C,(HL)                intrările în tabelul
                   INC   HL                    parametru si se aduce adresa
                   LD    B,(HL)                rutinei de comandă cerută.
                   EX    DE,HL                 Se schimbă indicatorii la loc
                   PUSH  BC                    si se face un salt indirect
                   RET                         la rutina de comandă.
```

THE 'COMMAND CLASSES - 01, 02 & 04' (CLASELE DE COMANDA 01, 02 & 04)
Aceste trei clase de comandă sînt folosite de comenzile care tratează variabilă - LET, FOR & NEXT si indirect de READ & INPUT.
Clasa de comandă 01 se preocupă de indentificarea variabilei într-o instructiune LET, READ sau INPUT.

```
1C22 CLASS-01      CALL  2882,LOOK-VARS        Se caută în zona variabilelor
                                               pentru a determina dacă cumva
                                               s-a folosit deja variabila.
```

THE 'VARIABILE IN ASSIGNMENT' SUBROUTINE (SUBRUTINA 'ATRIBUIRE VARIABILA')
Această subrutină dezvoltă valorile corespunzătoare pentru variabilele sistem DEST & STRLEN.

```
1C22 VAR-A-1       LD    (FLAGX),+00           Se initializează FLAGX cu +00
                   JR    NC,1C30,VAR-A-2       Salt înainte dacă variabila a
                                               mai fost folosită înainte.
                   SET   1,(FLAGX)             Semnalare 'variabilă nouă'.
                   JR    NZ1C46,VAR-A-3        Va apare eroare dacă se
                                               încearcă utilizarea unei
                                               'multimi nedimensionate'.
```

Prezentarea 2- Nu se găseste variabila

```
1C2E REPORT-2      RST   0008,ERROR-1           Se apelează rutina de tratare
                   DEFB  +01                   eroare.
```

Se continuă cu tratarea variabilelor existente.

```
1C30 VAR-A-2       CALL  Z,2996,STK-VARS       Parametrii unui sir simplu de
```

```
                                                  variabile si toata multimea
                                                  variabilelor sînt trecute în
                                                  stiva calculatorului.
                                                  (STK-VARS va 'tăia' un sir
                                                  dacă se cere.)
                BIT   6,(FLAGS)                   Salt înainte dacă se trateaza
                JR    NZ,1C46,VAR-A-3             o variabilă numerică.
                XOR   A                           Se sterge registrul A.
                CALL  2530,SYNTAX-Z               Parametrii sirului sau sirul
                CALL  NZ,2BF1,STK-FETCH           multimii variabile se aduc
                                                  numai dacă s-a verificat
                                                  sintaxa.
                LD    HL,+5C71                    Adresa FLAGX.
                OR    (HL)                        Bitul 0 este setat numai cînd
                LD    (HL),A                      se tratează un 'sir simplu'
                                                  astfel semnalizînd 'stergerea
                                                  copiei vechi'.
                EX    DE,HL                       Acum HL indică sirul sau
                                                  elementul multimii.
```

Traiectoriile se întîlnesc pentru setarea STRLEN & DEST după cum se cere. Pentru toate variabilele numerice si sirul 'nou' & multimea variabilelor sir STRLEN-lo contine 'litera' numelui variabilei. Dar pentru sirul 'vechi' & multimea variabilelor sir dacă sînt bucăti sau complete ea contine 'lungimea' în 'atribuire'.

```
1C46 VAR-A-3    LD    (STRLEN),BC                 Setare STRLEN cum se cere.
```

DEST contine adresa de 'destinatie a unei variabile 'vechi' dar de fapt 'sursa' pentru o variabilă 'nouă'.

```
                LD    (DEST),HL                   Setare DEST cum se cere si
                RET                               revenire.
```

Clasa de comandă 02 se preocupă cu calculul actual al valorii ce trebuie atribuite instructiunii LET.

```
1C4E CLASS-02   POP   BC                          Adresa SCAN-LOOP este trimisă
                CALL  1C56,VAL-FET-1              Se face atribuirea.
                CALL  1BEE,CHECK-END              Se trece la următoarea
                                                  instructiune fie prin
                                                  CHECK-END dacă s-a verificat
                RET                               sintaxa, fie STMT-RET, dacă
                                                  este 'perioada de executie'.
```

THE 'FETCH A VALUE' SUBRUTINE (SUBRUTINA 'ADUCEREA UNEI VALORI')
Această subrutină este folosită de instructiunile LET, READ & INPUT mai întîi pentru a evalua iar apoi pentru a atribui valoarea variabilei desemnate anterior.
Punctul de intrare VAL-FET-1 este folosit de LET & READ si ia în considerare FLAGS în timp ce punctul de intrare VAL-FET-2 este folosit de INPUT si ia în considerare FLAGX.

```
1C56 VAL-FET-1  LD    A,(FLAGS)                   Se foloseste FLAGS.
1C59 VAL-FET-2  PUSH  AF                          Se salvează FLAGS sau FLAGX.
                CALL  24FB,SCANNING               Se evaluează următoarea
                                                  expresie.
                POP   AF                          Se aduce vechea valoare a lui
                                                  FLAGS sau FLAGX.
                LD    D,(FLAGS)                   Se aduce noul FLAGS.
                XOR   D                           Natura - numeric sau sir, a
                AND   +40                         variabilei si a expresiei
                                                  trebuie să coincidă.
                JR    NZ,1C8A,REPORT-C            Se dă prezentarea C dacă ele
                                                  nu coincid.
                BIT   7,D                         Salt înainte pentru a realiza
                JP    NZ,2AFF,LET                 actuala atribuirea numai dacă
                RET                               s-a verificat sintaxa, cînd
                                                  se face o simplă revenire.
```

THE 'COMMAND CLASS 04' ROUTINE (RUTINA 'CLASA DE COMANDA 04')
Punctul de intrare al clasei de comandă 04 este folosit de instructiunile FOR & NEXT.

```
1C6C CLASS-04   CALL  28B2,LOOK-VARS              Se caută în zona variabilelor
                                                  dacă variabila a mai fost
                                                  utilizată.
                PUSH  AF                          Se salvează registrul pereche
                LD    A,C                         AF pînă cînd octetul
```

```
                OR    +9F                    discriminator este testat
                INC   A                      pentru a se asigura cà
                JR    NZ,1C8A,REPORT-C       variabila este o varibilà de
                                             control FOR-NEXT.
                POP   AF                     Se reface registrul
                JR    1C22,VAR-A-1           fanioanelor si salt înapoi
                                             pentru a face variabila care
                                             a fost gàsità 'variabilà în
                                             atribuire'.
```

THE 'EXPECT NUMERIC/STRING EXPRESSIONS' SUBROUTINE (SUBRUTINA 'ASTEPTARE EXPRESII NUMERICE/SIR')
Este o serie de subrutine scurte care sînt utilizate pentru a aduce rezultatul evaluàrii urmàtoarei expresii. Rezultatul dintr-o singurà expresie este returnat ca o 'ultimà valoare' în stiva calculatorului.
Punctul de intrare NEXT-2NUM este folosit cînd CH-ADD trebuie mărit pentru a indica începutul primei expresii.

```
1C79 NEXT-2NUM  RST   0020,NEXT-CHAR         Avans CH-ADD.
```

Punctul de intrare EXPT-2NUM (EQU. CLASS-08) permite evaluarea a două expresii numerice, separate prin virgulà.

```
1C7A EXPT-2NUM  CALL  1C82,EXPT-1NUM         Se evaluează fiecare expresie
     (CLASS-08)                              pe rînd - asa cà se evaluează
                                             prima.
                CP    +2C                    Se dà prezentarea de eroare
                JR    NZ,1C8A,REPORT-C       dacà separatorul nu este o
                                             virgulà.
                RST   0020,NEXT-CHAR         Avans CH-ADD.
```

Punctul de intrare EXPT-1NUM (EQU. CLASS-06) permite evaluarea unei singure expresii numerice.

```
1C82 EXPT-1NUM  CALL  24FB,SCANNING          Evaluarea urmàtoarei expresii
     (CLASS-06)
                BIT   6,(FLAGS)              Se revine atîta timp cît
                RET   NZ                     rezultatul este numeric;
                                             altfel este o eroare.
```

Prezentarea C - Nonsens în BASIC

```
1C8A REPORT-C   RST   0008,ERROR-1           Se apelează rutina de tratare
                DEFB  +0B                    eroare.
```

Punctul de intrare EXPT-EXP (EQU. CLASS-0A) permite evaluarea unei singure expresii sir.

```
1C8C EXPT-EXP   CALL  24FB,SCANNING          Se evaluează urmàtoarea
     (CLASS-0A)                              expresie.
                BIT   6,(FLAGS)              De aceastà datà se revine
                RET   Z                      dacà rezultatul indicà un
                JR    1C8A,REPORT-C          sir; altfel se prezintà
                                             eroare.
```

THE 'SET PERMANENT COLOURS' SUBROUTINE (EQU CLASS -07) (SUBRUTINA 'SETUL CULORILOR PERMANENTE' (EQU CLASS -07))
Aceastà subrutinà permite culorilor permanente temporare să devinà permanente. Ca si clasa de comandà 07, este de fapt o rutinà de comandà pentru comanda a sase culori.

```
1C96 PERMS      BIT   7,(FLAGS)              Fanionul sintaxà/executie
     (CLASS-07)                              este gata.
                RES   0,(TV-FLAG)            Semnalizare 'ecran principal'
                CALL  NZ,0D4D,TEMPS          Doar pe durata 'executiei' se
                                             apeleazà TEMPS pentru a
                                             asigura cà culoarea temporarà
                                             este culoarea ecranului
                                             principal.
                POP   AF                     Se trimite adresa de revenire
                                             SCAN-LOOP.
                LD    A,(T-ADDR)             Se aduce octetul inferior al
                                             lui T-ADDR si se scade +13
                SUB   +13                    pentru a da ordinul +D9 la
                                             +DE, care sînt codurile
                                             simbolului INK la OVER.
                CALL  21FC,CO-TEMP-4         Salt înainte pentru a schimba
                                             culorile temporare asa cum
                                             este indicat de instructiunea
```

```
                                                    BASIC.
                CALL  1BEE,CHECK-END                Se trece la următoarea
                                                    instructiune si se verifică
                                                    sintaxa.
                LD    HL,(ATTR-T)                   Acum valorile culorilor
                LD    (ATTR-P),HL                   temporare se fac permanente
                                                    (ATTR-P & MASK-P).
                LD    HL,+5C91                      Aceasta este P-FLAG; si
                LD    A,(HL)                        aceasta trebuie considerată.
```

Următoarele instructiuni copiază bitii pari ai octetului înlocuit cu bitii impari. De fapt realizarea bitilor permanenti este la fel ca si realizarea celor temporari.

```
                RLCA                                Mutare mască la stînga.
                XOR   8HL)                          Se imprimă pe mască numai
                AND   +AA                           bitii pari ai celuilalt octet
                XOR   (HL)
                LD    (HL),A                        Se reface rezultatul.
                RET
```

THE 'COMMAND CLASS 09' ROUTINE (RUTINA 'CLASA DE COMANDA 09')
Această rutină este folosită de instructiunile PLOT, DRAW & CIRCLE în ordine pentru a specifica absenta conditiilor 'FLASH 8; BRIGHT 8; PAPER 8'; acestea sînt setate înaintea considerării oricărui număr de culoare introdus.

```
1CBE CLASS-09   CALL  2530,SYNTAX-Z                 Salt înainte dacă sintaxa s-a
                JR    Z,1CD6,CL-09-1                verificat.
                RES   0,(TV-FLAG)                   Semnalizare 'ecran principal'
                CALL  0D4D,TEMPS                    Setare culori temporare
                                                    pentru ecranul principal.
                LD    HL,+5C90                      Aceasta este MASK-T.
                LD    A,(HL)                        Se aduce valoarea ei curentă
                OR    +F8                           dar se retine numai partea
                                                    'nemascată' a lui INK.
                LD    (HL),A                        Se reface valoarea, care
                                                    acum indică 'FLASH 8;
                                                    BRIGHT 8; PAPER 8'.
                RES   6,(P-FLAG)                    De asemenea se asigură că nu
                                                    este 'PAPER 9'.
                RST   0018,GET-CHAR                 Se aduce caracterul prezent
                                                    înainte de a se confirma
                                                    lucrul cu numărul culorii
                                                    introduse.
1CD6 CL-09-1    CALL  21E2,CO-TEMP                  Se lucrează cu numărul
                                                    culorii locale dominante.
                JR    1C7A,EXPT-2NUM                Acum se aduc primii doi
                                                    operanzi pentru PLOT, DRAW
                                                    sau CIRCLE.
```

THE 'COMMAND CLASS 0B' ROUTINE (RUTINA 'CLASA DE COMANDA 0B')
Această rutină este folosită este instructiunile SAVE, LOAD, VERIFY & MERGE.

```
1CDB CLASS-0B   JP    0605,SAVE-ETC                 Salt la rutina de tratare a
                                                    casetei.
```

THE 'FETCH A NUMBER' SUBROUTINE (SUBRUTINA 'ADUCEREA UNUI NUMAR')
Această subrutină conduce la următoarea expresie numerică ce trebuie evaluată, dar în cazul în care nu este nici o expresie se foloseste zero.

```
1CDE FETCH-NUM  CP    +0D                           Salt înainte dacă s-a ajuns
                JR    Z,1CE6,USE-ZERO               la sfîrsitul liniei.
                CP    +3A                           Dar salt la EXPT-1NUM dacă nu
                JR    NZ,1C82,EXPT-1NUM             este sfîrsitul instructiei.
```

Acum calculatorul este folosit pentru aducerea valorii zero în stiva calculatorului.

```
1CE6 USE-ZERO   CALL  2530,SYNTAX-Z                 Nu se prelucrează operatia
                RET   Z                             dacă se verificat sintaxa.
                RST   0028,FP-CALC                  Utilizare calculator.
                DEFB  +A0,stk-zero                  'Ultima valoare' este acum
                DEFB  +38,end-calc                  zero.
                RET                                 Revenire cu zero adăugat la
                                                    stivă.
```

THE COMMAND ROUTINES (RUTINELE DE COMANDA)
Sectiunea celor 16K de program monitor de la 1CEE la 23FA contine cele mai multe dintre rutinele de comandă ale interpretorului BASIC.

THE 'STOP' COMMAND ROUTINE (RUTINA COMANDA 'STOP')
Rutina comandă pentru STOP contine doar un apel al rutinei de tratare a erorii.

```
1CEE STOP       RST   0008,ERROR-1        Se apelează rutina de tratare
     (REPORT-9) DEFB  +08                 eroare.
```

THE 'IF' COMMAND ROUTINE (RUTINA COMANDA 'IF')
La intrare, valoarea expresiei între IF si THEN este 'ultima valoare' din stiva calculatorului. Dacă este adevărat logic, atunci se ia în considerare următoarea instructie; în caz contrar se consideră că linia s-a terminat.

```
1CF0 IF         POP   BC                 Se trimite adresa de revenire
                                         STMT-RET.
                CALL  2530,SYNTAX-Z      Salt înainte dacă se verifică
                JR    Z,1D00,IF-1        sintaxa.
```

Acum se foloseste calculatorul pentru a 'sterge' ultima valoare din stiva calculatorului dar se lasă registrul pereche DE să adreseze primul octet al valorii.

```
                RST   0028,FP-CALC       Utilizare calculator.
                DEFB  +02,delete         Prezenta 'ultima valoare' se
                DEFB  +38,end-calc       sterge.
                EX    DE,HL              HL indică primul octet si
                CALL  34E9,TEST-ZERO     apelează TEST-ZERO.
                JP    C,1B83,LINE-END    Dacă valoarea a fost 'FALSE'
                                         (fals) salt la linia
                                         următoare.
1D00 IF-1       JP    1B29,STMT-L-1      Dacă a fost 'TRUE'(adevărată)
                                         se sare la următoarea
                                         instructiune (după THEN).
```

THE 'FOR' COMMAND ROUTINE (RUTINA DE COMANDA 'FOR')
Această rutină de comandă este introdusă cu VALUE (valoarea) si LIMIT (limita) pentru instructiunea FOR aflate deja în vîrful stivei calculatorului.

```
1D03 FOR        CP    +CD                Salt înainte dacă nu s-a dat
                JR    NZ,1D10,F-USE-1    un 'STEP' (pas).
                RST   0020,NEXT-CHAR     Avans CH-ADD si se aduce
                CALL  1C82,EXPT-1NUM     valoarea pasului.
                CALL  1BEE,CHECK-END     Se trece la următoarea
                JR    1D16,F-REORDER     instructiune dacă se verifică
                                         sintaxa;altfel, salt înainte.
```

Nu s-a dat valoarea pentru STEP (pas), asa că se va folosi valoarea '1'.

```
1D10 F-USE-1    CALL  1BEE,CHECK-END     Se trece la următoarea
                                         instructie dacă se verifică
                                         sintaxa; altfel se foloseste
                RST   0028,FP-CALC       calculatorul pentru a pune
                DEFB  +A1,stk-one        '1' în stiva calculatorului.
                DEFB  +38,end-calc
```

Cele trei valori din stiva calculatorului sînt VALUE (v) (valoare), LIMIT (l) (limita) si STEP (s) (pas). Acum aceste valori trebuie modificate.

```
1D16 F-REORDER  RST   0028,FP-CALC       v,l,s
                DEFB  +C0,st-mem-0       v,l,s          (mem-0 = s)
                DEFB  +02,delete         v,l
                DEFB  +01,exchange       l,v
                DEFB  +E0,get-mem-0      l,v,s
                DEFB  +01,exchange       l,v,s
                DEFB  +38,end-calc
```

Acum se stabileste o variabilă de control FOR si se tratează ca o zonă de memorie temporară a calculatorului.

```
                CALL  2AFF,LET           Variabila este găsită sau
                                         creeată dacă este necesar
                                         (este folosit v).
                LD    (MEN),HL           Se face 'zonă de memorie'.
```

Variabila care s-a găsit poate fi o variabilă numerică simplă care foloseste doar sase locatii, caz în care este necesară extensia.

```
                DEC   HL                    Se aduce singurul caracter
                LD    A,(HL)                al numelui variabilei.
                SET   7,(HL)                Se asigurà cà bitul 7 al
                                            numelui este setat.
                LD    BC,+0006              Vor fi sase locatii la
                                            sfîrsit.
                ADD   HL,BC                 HL indicà dupà ele.
                RLCA                        Se roteste numele si salt
                JR    C,1D34,F-L&S          dacà a fost deja o variabilà
                                            FOR.
                LD    C,+0D                 Altfel, se mai creeazà 13
                CALL  1655,MAKE-ROOM        locatii.
                INC   HL                    Din nou HL va indica pozitia
                                            LIMIT.
```

Acum se adaugà valorile initiale ale lui LIMIT si STEP.

```
1D34 F-L&S      PUSH  HL                    Este salvat indicatorul.
                RST   0028,FP-CLAC          1,s
                DEFB  +02,delete            1
                DEFB  +02,delete            -
                DEFB  +38,end-calc          DE indicà încà pe '1'.
                POP   HL                    Indicatorul este refàcut si
                EX    DE,HL                 cei doi indicatori sînt
                                            interschimbati.
                LD    C,+0A                 Se mutà cei zece octeti ai
                LDIR                        lui LIMIT si STEP.
```

Se introduc acum ciclàri pentru numàrul liniei si numàrul instructiunii.

```
                LD    HL,(PPC)              Numàrul liniei curente.
                EX    DE,HL                 Se interschimbà registrii
                LD    (HL),E                înainte de a adàuga numàrul
                INC   HL                    liniei la variabila de
                LD    (HL),D                control FOR.
                LD    D,(SUBPPC)            Instructiunea de ciclare este
                INC   D                     întotdeauna urmàtoarea
                INC   HL                    instructiune - dacà existà
                LD    (HL),D                sau nu.
```

Subrutina NEXT-LOOP este apelatà pentru a testa posibilitatea unei 'treceri' si se revine dacà existà una posibilà; altfel instructiunea dupà bucla FOR - NEXT trebuie identificatà.

```
                CALL  1DDA,NEXT-LOOP        Este 'o trecere' posibilà?
                RET   NC                    Revenire dacà nu este.
                LD    B,(STRLEN-lo)         Se aduce numele variabilei.
                LD    HL,(PPC)              Se copiazà numele liniei
                LD    (NEWPPC),HL           prezente în PPC.
                LD    A,(SUBPPC)            Se aduce numàrul instructiei
                NEG                         curente si se complementeazà
                                            fatà de 2.
                LD    D,A                   Se transferà rezultatul în
                                            registrul D.
                LD    HL,(CH-ADD)           Se aduce valoarea curentà a
                                            lui CH-ADD.
                LD    E,+F3                 Se cautà 'NEXT'.
```

Acum se cautà în zona de program, de la indicatorul prezent în continuare, prima interventie a lui NEXT urmatà de o variabilà corectà.

```
1D64 F-LOOP     PUSH  BC                    Se salveazà numele variabilei
                LD    BC,(NXTLIN)           Se aduce valoarea curentà a
                                            lui NXTLIN.
                CALL  1D86,LOOK-PROG        Acum este cercetat spatiul
                                            programului si BC va fi
                                            schimbat cu fiecare linie
                                            cercetatà.
                LD    (NXTLIN),BC           Se salveazà indicatorul.
                POP   BC                    Se reface numele variabilei.
                JR    C,1D84,REPORT-1       Dacà mai departe nu existà
                                            NEXT, atunci se trimite
                                            eroare.
                RST   0020,NEXT-CHAR        Se avanseazà peste NEXT care
                                            a fost gàsit.
                OR    +20                   Se permite despàrtirea
                CP    B                     literelor mici de literele
```

```
                                        mari  înaintea  testării
                                        numelui noii variabile.
        JR    Z,1D7C,F-FOUND            Salt inainte dacă coincid.
        RST   0020,NEXT-CHAR            Din nou avans CH-ADD si salt
        JR    1D64,F-LOOP               înapoi  dacă  nu este corectă
                                        variabila.
```

NEWPPC conține numărul de linie al liniei în care s-a găsit NEXT corect. Acum trebuie găsit numărul instructiunii si stocat în NSPPC.

```
1D7C F-FOUND    RST   0020,NEXT-CHAR    Adresa CH-ADD.
                LD    A,+01             Contorul de instructiuni din
                SUB   D                 registrul  D  numără
                                        instructiunile înapoi de la
                                        zero, asa că trebuie scăzut
                                        cu '1'.
                LD    (NSPPC),A         Rezultatul este stocat.
                RET                     Acum revenire - la STMT-RET.
```

Prezentarea I - FOR fără NEXT

```
1D84 REPORT-1   RST   0008,ERROR-1      Se apelează rutina de tratare
                DEFB  +11               eroare.
```

THE 'LOOK-PROG' SUBROUTINE (SUBRUTINA 'CAUTA PROGRAM')
Această subrutină este folosită pentru a găsi interventiile pentru DATA, DEF FN sau NEXT. La intrare codul simbolului corespunzător este în registrul E iar registrul pereche HL indică începutul zonei în care se caută.

```
1D86 LOOK-PROG  LD    A,(HL)            Se aduce prezentul caracter.
                CP    +3A               Salt înainte dacă este ':'
                JR    Z,1DA3,LOOK-P-2   care va indica faptul că mai
                                        sînt instructiuni în linia
                                        prezentă.
```

Acum se introduce o buclă care va examina mai departe fiecare linie din program.

```
1D8B LOOK-P-1   INC   HL                Se aduce octetul superior al
                LD    A,(HL)            numărului liniei si se revine
                AND   +C0               cu transportul setat dacă nu
                SCF                     mai sînt linii în program.
                RET   NZ
                LD    B,(HL)            Se aduce numărul liniei si se
                INC   HL                trece în NEWPPC.
                LD    C,(HL)
                LD    (NEWPPC),BC
                INC   HL                Apoi este colectată lungimea.
                LD    C,(HL)
                INC   HL
                LD    B,(HL)
                PUSH  HL                Indicatorul este salvat pînă
                ADD   HL,BC             cînd adresa sfîrsitului de
                LD    B,H               linie  este  formată  în
                LD    C,L               registrul pereche BC.
                POP   HL                Se reface indicatorul.
                LD    D,+00             Se  setează  contorul
                                        instructiunilor pe zero.
1DA3 LOOK-P-2   PUSH  BC                Indicatorul sfîrsit-de-linie
                CALL  198B,EACH-STMT    este  salvat  pînă cînd sînt
                POP   BC                examinate instructiile de pe
                                        linie.
                RET   NC                Se  revine  dacă  a  fost  o
                JR    1D8B,LOOK-P-1     'interventie';  altfel  se
                                        consideră următoarea linie.
```

THE 'NEXT' COMMAND ROUTINE (RUTINA COMANDA 'URMATOAREA')
'Atribuirea valorii' a fost deja determinată (vezi CLASS -04, 1C6C); a rămas să se schimbe VALUE (valoarea) asa cum se cere.

```
1DAB NEXT       BIT   1,(FLAGX)         Salt pentru a prezenta eroare
                JP    NZ,1C2E,REPORT-2  dacă nu s-a găsit variabila.
                LD    HL,(DEST)         Se aduce adresa variabilei si
                BIT   7,(HL)            apoi se testează numele.
                JR    Z,1DD8,REPORT-1
```

In continuare calculatorul prelucrează valoarea (VALUE) si pasul (STEP) variabilei.

```
                INC   HL                    Se trece peste nume.
                LD    (MEM),HL              Face variabila o 'zona de
                                            memorie' temporară.
                RST   0028,FP-CALC          -
                DEFB  +E0,get-mem-0         v
                DEFB  +E2,get-mem-2         v,s
                DEFB  +0F,addition          v+s
                DEFB  +C0,st-mem-0          v+s
                DEFB  +02,delete            -
                DEFB  +38,end-calc          -
```

Rezultatul obtinut prin adunarea lui VALUE si STEP este acum comparat cu LIMIT apelînd NEXT-LOOP.

```
                CALL  1DDA,NEXT-LOOP        Se compară noua VALUE fată de
                                            LIMIT.
                RET   C                     Acum se revine dacă ciclul
                                            FOR-NEXT a fost complet.
```

Altfel se adună 'ciclarea' numărului de linie si instructiei.

```
                LD    HL,(MEM)              Se caută adresa octetului
                LD    DE,+000F              inferior a numărului liniei
                ADD   HL,DE                 ciclate.
                LD    E,(HL)                Acum se aduce numărul acestei
                                            linii.
                INC   HL
                LD    D,(HL)
                INC   HL
                LD    H,(HL)                Urmează numărul instructiei.
                EX    DE,HL                 Se schimbă numerele înainte
                JP    1E73,GO-TO-2          de a sări în fată pentru a fi
                                            tratate ca linia de
                                            destinatie a comenzii GO TO.
```

Prezentarea 1 - NEXT fără FOR

```
1DD8 REPORT-1   RST   0008,ERROR-1          Se apelează rutina de tratare
                DEFB  +00                   eroare.
```

THE 'NEXT-LOOP' SUBROUTINE (SUBRUTINA 'URMATOAREA BUCLA')
Această buclă este folosită pentru a determina dacă LIMIT a fost depăsită de prezenta VALUE. Nota trebuie luată din semnul lui STEP.
Subrutina revine cu fanionul de transport setat dacă LIMIT este depăsită.

```
1DDA NEXT-LOOP  RST   0028,FP-CALC          -
                DEFB  +E1,get-mem-1         1
                DEFB  +E0,get-mem-0         1,v
                DEFB  +E2,get-mem-2         1,v,s
                DEFB  +36,less-0            1,v,(1/0)
                DEFB  +00,jump-true         1,v,(1/0)
                DEFB  +02,to NEXT-1         1,v,(1/0)
                DEFB  +01,exchange          v,1
1DE2 NEXT-1     DEFB  +03,subtract          v-1 sau 1-v
                DEFB  +37,greater-0         (1/0)
                DEFB  +00,jump-true         (1/0)
                DEFB  +04,to NEXT-2         -
                DEFB  +38,end-calc          -
                AND   A                     Se sterge fanionul de
                RET                         transport si revenire -
                                            ciclare posibilă.
```

Dacă ciclarea este imposibilă fanionul de transport este setat.

```
1DE9 NEXT-2     DEFB  +38,end-calc          -
                SCF                         Setare fanion de transport si
                RET                         revenire.
```

THE 'READ' COMMAND ROUTINE (RUTINA DE COMANDA 'READ')
Comanda READ (citire) permite citirea unei liste DATA si are un efect similar cu o serie de instructiuni LET.
Fiecare atribuire în cadrul unei singure instructiuni READ este tratată una după alta. Variabila sistem X-PTR este utilizată ca o locatie de stocare a indicatorului instructiunii READ pînă cînd CH-ADD este utilizat pentru a parcurge lista DATA.

```
1DEC READ-3     RST   0020,NEXT-CHAR        Aici se vine la fiecare
                                            trecere, după prima, pentru
                                            mutare de-a lungul
```

```
                                                    instructiunii READ.
1DED READ       CALL  1C1F,CLASS-01                 Se ia în cosiderare dacă
                                                    variabilele au mai fost
                                                    folosite înainte; se caută
                                                    intrarea existentă, dacă a
                                                    fost.
                CALL  2530,SYNTAX-Z                 Salt înainte dacă se verifică
                JR    Z,1E1E,READ-2                 sintaxa.
                RST   0018,GET-CHAR                 Se salvează indicatorul
                LD    (X-PTR),HL                    curent CH-ADD în X-PTR.
                LD    HL,(DATADD)                   Se aduce indicatorul curent
                LD    A,(HL)                        al listei DATA si salt
                CP    +2C                           înainte dacă nu se mai află o
                JR    Z,1E0A,READ-1                 nouă instructiune DATA.
                LD    E,+E4                         Căutarea este pentru 'DATA'.
                CALL  1D86,LOOK-PROG                In cazul în care căutarea a
                JR    NC,1E0A,READ-1                reusit, salt înainte.

Prezentarea E - Afară din DATA

1E08 REPORT-E   RST   0008,ERROR-1                  Se apelează rutina de tratare
                DEFB  +00                           eroare.

Se continuă - se scoate o valoare din lista DATA.

1E0A READ-1     CALL  0077,TEMP-PTR1                Indicatorul avansează de-a
                                                    lungul listei DATA si setare
                                                    CH-ADD.
                CALL  1C56,VAL-FET-1                Se aduce valoarea si se i se
                                                    atribuie variabilei.
                RST   0018,GET-CHAR                 Se aduce valoarea curentă a
                LD    (DATADD),HL                   lui CH-ADD si se stochează în
                                                    DATADD.
                LD    HL,(X-PTR)                    Se aduce indicatorul la
                LD    (X-PTR-hi),+00                instructiunea READ si se
                                                    sterge X-PTR.
                CALL  0078,TEMP-PTR2                CH-ADD va indica încă o dată
                                                    instructiunea READ.
1E1E READ-2     RST   0018,GET-CHAR                 Se aduce prezentul caracter
                CP    +2C                           si se vede dacă este ','.
                JR    Z,1DEC,READ-3                 Dacă este, atunci salt înapoi
                                                    ca si cum ar urma numere;
                CALL  1BEE,CHECK-END                altfel, se revine fie prin
                RET                                 CHECK-END (dacă se verifică
                                                    sintaxa), fie cu instructia
                                                    RET (la STMT-RET).
```

THE 'DATA' COMMAND ROUTINE (RUTINA DE COMANDA 'DATA')
In timpul verificării sintaxei, se verifică o instructiune DATA pentru a se asigura că ea contine o serie de expresii valide, separate prin virgulă. Dar în 'timpul executiei' această instructiune este omisă.

```
1E27 DATA       CALL  2530,SYNTAX-Z                 Salt înainte dacă se
                JR    NZ,1E37,DATA-2                verifică sintaxa.
```

Acum se introduce o buclă care lucrează cu fiecare expresie din instructiunea DATA.

```
1E2C DATA-1     CALL  24FB,SCANNING                 Defilarea expresiei următoare
                CP    +2C                           Se verifică dacă separatorul
                                                    este corect - o ',';
                CALL  NZ,1BEE,CHECK-END             însă se trece la următoarea
                                                    instructiune dacă nu coincid.
                RST   0020,NEXT-CHAR                Bucla se ciclează până când
                JR    1E2C,DATA-1                   mai sînt expresii ce trebuie
                                                    verificate.
```

Instructiunea DATA trebuie omisă în 'timpul executiei'.

```
1E37 DATA-2     LD    A,+E4                         Este o instructie DATA care
                                                    trebuie omisă.
```

THE 'PASS-BY' SUBROUTINE (SUBRUTINA 'TRECERE' 'OMITERE')
La intrare registrul A va contine fie simbolul 'DATA' fie simbolul 'DEF FN', în functie de instructiunea care a fost 'sărită'.

```
1E39 PASS-BY    LD    B,A                           Registrul pereche BC va
                                                    contine un număr foarte mare.
                CPDR                                Se caută înapoi de-a lungul
```

```
                        LD    DE,+0200          instructiunii simbolul.
                        JP    198B,EACH-STMT    Acum caută de-a lungul liniei
                                                de după instructiune.(A 'D-1'
                                                instructie de la pozitia
                                                curentă.)
```

THE 'RESTORE' COMMAND ROUTINE (RUTINA DE COMANDA 'RESTORE')
Operandul unei instructiuni RESTORE este luat ca un număr de linie, iar în cazul în care nu este dat nici un operand se consideră zero.
Punctul de intrare REST-RUN este folosit de rutina de comandă RUN.

```
1E42 RESTORE        CALL  1E99,FIND-INT2      Se comprimă operandul în
                                              registrul pereche BC.
1E45 REST-RUN       LD    H,B                 Rezultatul este transferat în
                    LD    L,C                 registrul pereche HL.
                    CALL  196E,LINE-ADDR      Acum se determină adresa
                                              acelei linii sau 'a primei
                                              linii de după'.
                    DEC   HL                  DATADD va indica locatia de
                    LD    (DATADD),HL         dinainte.
                    RET                       Revenire odată ce s-a făcut.
```

THE 'RANDOMIZE' COMMAND ROUTINE (RUTINA DE COMANDA 'RANDOMIZE')
Din nou operandul este comprimat în registrul pereche BC si transferat în variabila sistem cerută. Dacă operandul este zero, imediat se utilizează valoarea din FRAMES1 si FRAMES2.

```
1E4F RANDOMIZE      CALL  1E99,FIND-INT2      Se aduce operandul.
                    LD    A,B                 Salt înainte dacă valoarea
                    OR    C                   operandului este zero.
                    JR    NZ,1E5A,RAND-1
                    LD    BC,(FRAMES1)        Se aduc imediat cei doi
                                              octeti de ordin inferior din
                                              FRAMES.
1E5A RAND-1         LD    (SEED),BC           Inainte de revenire se
                    RET                       introduce rezultatul în
                                              variabila sistem SEED.
```

THE 'CONTINUE' COMMAND ROUTINE (RUTINA DE COMANDA 'CONTINUARE')
Numărul de linie cerut si numărul instructiunii din acea linie care fac obiectul unui salt.

```
1E5F CONTINUE       LD    HL,(OLDPPC)         Numărul liniei.
                    LD    D,(OSPCC)           Numărul instructiunii.
                    JR    1E73,GO-TO-2        Salt înainte.
```

THE 'GO TO' COMMAND ROUTINE (RUTINA DE COMANDA 'GO TO')
Operandul unui GO TO trebuie să fie un număr de linie în limitele '1' la '9999', dat actualul test se face pentru o valoare mai mare de '61439'.

```
1E67 GO-TO          CALL  1E99,FIND-INT2      Se aduce operandul si se
                    LD    H,B                 transferă în registrul
                                              pereche HL.
                    LD    L,C
                    LD    D,+00               Setare număr instructiune cu
                                              zero.
                    LD    A,H                 Se dă mesaj de eroare -
                    CP    +F0                 partea întreagă iese din
                    JR    NC,1E9F,REPORT-B    limite - cu linii peste
                                              '61439'.
```

Punctul de intrare GO-TO-2 este folosit la determinarea numărului liniei a liniei următoare de tratat în cîteva situatii.

```
1E73 GO-TO-2        LD    (NEWPPC),HL         Se introduce numărul liniei
                    LD    (NSPPC),D           si apoi numărul instructiei.
                    RET                       Revenire; - la STMT-RET.
```

THE 'OUT' COMMAND ROUTINE (RUTINA DE COMANDA 'IESIRE')
Cei doi parametrii ai instructiunii OUT sînt adusi din stiva calculatorului si folositi după indicatii.

```
1E7A OUT            CALL  1E85,TWO-PARAM      Se aduc parametrii.
                    OUT   (C),A               Instructia OUT actuală.
                    RET                       Revenire; - la STMT-RET.
```

THE 'POKE' COMMAND ROUTINE (RUTINA DE COMANDA 'INTRODUCERE')
Intr-o manieră similară se execută si operatiunea POKE.

```
1E80 POKE       CALL  1E85,TWO-PARAM     Se aduc operanzii.
                LD    (BC),A             Operatia POKE actuală.
                RET                      Revenire; - la STMT-RET.
```

THE 'TWO-PARAM' SUBROUTINE (SUBRUTINA 'DOI-PARAM')
Parametrul din vîrful stivei calculatorului trebuie comprimat ca să intre într-un singur registru. Dacă este negativ, se complementează față de doi. Al doilea parametru trebuie să fie comprimabil într-un registru pereche.

```
1E85 TWO-PARAM  CALL  2DD5,FP-TO-A       Se aduce parametrul.
                JR    C,1E9F,REPORT-B    Se dă eroare dacă numărul
                                         este prea mare.
                JR    Z,1E8E,TWO-P-1     Salt înainte în cazul
                NEG                      numerelor pozitive, dar dacă
                                         numărul este negativ se face
                                         complementul față de doi.
1E8E TWO-P-1    PUSH  AF                 Se salvează primul parametru
                CALL  1E99,FIND-INT2     pînă cînd se aduce cel de al
                                         doilea parametru.
                POP   AF                 Se restituie primul parametru
                RET                      înainte de revenire.
```

THE 'FIND INTEGERS' SUBROUTINES (SUBRUTINA 'GASIRE PARTE INTREAGA')
'Ultima valoare' din stiva calculatorului este adusă si comprimată într-un singur registru sau într-un registru pereche cu intrarea pe la FIND-INT1 si, respectiv, FIND-INT2.

```
1E94 FIND-INT1  CALL  2DD5,FP-TO-A       Se aduce 'ultima valoare'.
                JR    1E9C,FIND-I-1      Salt înainte.
1E99 FIND-INT2  CALL  2DA2,FP-TO-BC      Se aduce 'ultima valoare'.
1E9C FIND-I-1   JR    C,1E9F,REPORT-B    In ambele cazuri depăsirea
                RET   Z                  este indicată prin setarea
                                         fanionului de transport.
```

Prezentarea B - Parte întreagă afară din interval.

```
1E9F REPORT-B   RST   0008,ERROR-1       Se apelează rutina de tratare
                DEFB  +0A                eroare.
```

THE 'RUN' COMMAND ROUTINE (RUTINA DE COMANDA 'EXECUTIE')
Parametrul comenzii RUN este trecut în NEWPPC prin apelarea rutinei de comanadă GO TO. Se execută apoi operatiile 'RESTORE 0' si 'CLEAR 0' înainte de a se reveni.

```
1EA1 RUN        CALL  1E67,GO-TO         Setare NEWPPC după cum este
                                         cerut.
                LD    BC,+0000           Executie 'RESTORE 0'.
                CALL  1E45,REST-RUN
                JR    1EAF,CLEAR-1       Iesire prin rutina de comandă
                                         CLEAR.
```

THE 'CLEAR' COMMAND ROUTINE (RUTINA DE COMANDA 'CURATIRE')
Această rutină permite curătirea zonei variabilelor, ecranul este curătat si RAMTOP transferat. Ca o consecintă a ultimei operatii stiva calculatorului este refăcută, aceasta având ca efect si curătirea stivei GO SUB.

```
1EAC CLEAR      CALL  1E99,FIND-INT2     Se aduce operandul - în caz
                                         că lipseste se foloseste
                                         zero.
1EAF CLEAR-RUN  LD    A,B                Salt înainte dacă operandul
                OR    C                  nu este zero. Cînd se
                JR    NZ,1EB7,CLEAR-1    apelează din RUN nu se face
                                         salt.
                LD    BC,(RAMTOP)        Dacă este zero se foloseste
                                         valoarea existentă în RAMTOP.
1EB7 CLEAR-1    PUSH  BC                 Salvare valoare.
                LD    DE,(VARS)          Urmează refacerea tuturor
                LD    HL,(E-LINE)        octetilor din prezenta zonă
                DEC   HL                 a variabilelor.
                CALL  19E5,RECLAIM-1
                CALL  0D6B,CLS           Se curătă spatiul ecranului.
```

Se testează valoarea din registrul pereche BC ce va fi folosită ca RAMTOP pentru a se asigura că nu este nici prea mare, nici prea mică.

```
                LD    HL,(STKEND)        Se incrementează valoarea
                LD    DE,+0032           curentă a lui STKEND cu '50'
                ADD   HL,DE              înainte de a fi testată.
                POP   DE                 Aceasta va fi limita
```

```
                SBC   HL,DE             inferioară.
                JR    NC,1EDA,REPORT-M  RAMTOP va fi prea jos.
                LD    HL,(P-RAMT)       Pentru testul superior
                AND   A                 valoarea lui RAMTOP este
                SBC   HL,DE             comparată cu P-RAMT.
                JR    NC,1EDC,CLEAR-2   Salt înainte dacă este
                                        acceptată.
```

Prezentarea M - RAMTOP nu este bun

```
1EDA REPORT-M   RST   0008,ERROR-1      Se apelază rutina de tratare
                DEFB  +15               eroare.
```

Se continuă cu operatia CLEAR (stergere)

```
1EDC CLEAR-3    EX    DE,HL             Acum valoarea poate fi
                LD    (RAMTOP),HL       trecută în RAMTOP.
                POP   DE                Se aduce adresa - STMT-RET.
                POP   BC                Se aduce 'adresa erorii'.
                LD    (HL),+3E          Se introduce marcatorul de
                                        sfîrsit stivă GO SUB.
                DEC   HL                Se lasă o locatie.
                LD    SP,HL             Indicatorul stiva va indica
                                        stiva GO SUB goală.
                PUSH  BC                Apoi se trece 'adresa erorii'
                LD    (ERR-SP),SP       în stivă si se salvează
                                        adresa la ERR-SP.
                EX    DE,HL             Se face acum o revenire
                JP    (HL)              indirectă la STMT-RET.
```

Notă: Cînd rutina este apelată de RUN valorile lui NEWPPC & NSPPC vor fi afectate si nici o instructiune care apare după RUN nu va putea fi găsită înainte de a se efectua saltul.

THE 'GO SUB' COMMAND ROUTINE (RUTINA DE COMANDA 'GO SUB')
Prezenta valoare a lui PPC si valoarea incrementată a lui SUBPPC sînt stocate în stiva GO SUB.

```
1EED GO-SUB     POP   DE                Se salvează adresa - STMT-RET
                LD    H,(SUBPPC)        Se aduce numărul instructiei
                INC   H                 si se incrementează.
                EX    (SP),HL           Se schimbă 'adresă eroare' cu
                                        numărul instructiei.
                INC   SP                Se reface utilizarea unei
                                        locatii.
                LD    BC,(PPC)          Apoi se salvează numărul
                PUSH  BC                liniei prezente.
                PUSH  HL                Restituire 'adresă eroare' în
                LD    (ERR-SP),SP       stiva calculatorului si se
                                        resetează ERR-SP pentru a o
                                        indica.
                PUSH  DE                Restituire adresă - STMT-RET.
                CALL  1E67,GO-TO-1      Acum se setează NEWPPC &
                                        NSPPC la valorile cerute.
                LD    BC,+0014          Dar înainte de a face saltul
                                        se face un test pentru
                                        locatie.
```

THE 'TEST-ROOM' SUBROUTINE (SUBRUTINA 'TEST CAMERA')
Se execută o serie de teste pentru a se asigura că există suficientă memorie liberă accesibilă pentru taskul care a fost preluat.

```
1F05 TEST-ROOM  LD    HL,(STKEND)       Se incrementează valoarea
                ADD   HL,BC             luată din STKEND cu valoarea
                                        transportată în rutină prin
                                        registrul pereche BC.
                JR    C,1F15,REPORT-4   Salt înainte dacă rezultatul
                                        este peste +FFFF.
                EX    DE,HL             Se încearcă din nou,
                LD    HL,+0050          acceptînd următorii opt
                ADD   HL,DE             octeti.
                JR    C,1F15,REPORT-4
                SBC   HL,SP             In final valoarea se compară
                                        cu adresa din stiva
                                        calculatorului.
                RET   C                 Dacă rezultatul comparării
                                        este satisfăcător, se revine.
```

Prezentarea 4 - Afară din memorie

```
1F15 REPORT-4     LD    L,+03              Este o eroare 'timp de
                  JP    0055,ERROR-3       executie' si marcatorul de
                                           eroare nu este folosit.
```

THE 'FREE MEMORY' SUBROUTINE (SUBRUTINA 'MEMORIE LIBERA')
Nu există nici o comandă BASIC 'FREE' în SPECTRUM dar este o subrutină care realizează acest lucru.
O estimare a cantității de spatiu liber se poate găsi oricînd utilizînd:
'PRINT 65536-USR 7962'

```
1F1A FREE-MEM     LD    BC,+0000           Nu se admite orice deasupra.
                  CALL  1F05,TEST-ROOM     Se face testul si se trece
                  LD    B,H                rezultatul în registrul BC
                  LD    C,L                înainte de a se reveni.
                  RET
```

THE 'RETURN' COMMAND ROUTINE (RUTINA DE COMANDA 'RETURN')
Numărul liniei si numărul instructiunii care fac obiectul unui 'RETURN' sînt aduse din stiva GO SUB.

```
1F23 RETURN       POP   BC                 Se aduce adresa - STMT-RET.
                  POP   HL                 Se aduce 'adresă eroare'.
                  POP   DE                 Se aduce ultima intrare în
                                           stiva GO SUB.
                  LD    A,D                Intrarea se testează pentru a
                  CP    +3E                vedea dacă este marcatorul de
                  R     Z,1F36,REPORT-7    sfîrsit al stivei GO SUB;
                                           salt dacă este.
                  DEC   SP                 Intreaga intrare foloseste
                                           numai trei locatii.
                  EX    (SP),HL            Se schimbă numărul
                                           instructiei cu 'adresa
                                           eroare'.
                  EX    DE,HL              Se mută numărul instructiei.
                  LD    (ERR-SP),SP        Se resetează indicatorul de
                                           eroare.
                  PUSH  BC                 Se înlocuieste adresa -
                                           STMT-RET.
                  JP    1E73,GO-TO-2       Salt înapoi pentru a schimba
                                           NEWPPC & NSPPC.
```

Prezentarea 7 - RETURN fără GO SUB

```
1F36 REPORT-7     PUSH  DE                 Se înlocuieste marcatorul de
                  PUSH  HL                 sfîrsit si 'adresa eroare'.
                  RST   0008,ERROR-1       Se apelează rutina de tratare
                  DEFB  +06                eroare.
```

THE 'PAUSE' COMMAND ROUTINE (RUTINA DE COMANDA 'PAUSE')
Perioada unei PAUSE este determinată prin contorizarea numărului de întreruperi mascabile asa cum intervin la fiecare 1/50 dintr-o secundă.
O PAUSE se termină fie după numărul corespunzător de întreruperi fie prin variabila sistem FLAGS care indică faptul că s-a apăsat o tastă.

```
1F3A PAUSE        CALL  1E99,FIND-INT2     Se aduce operandul.
1F3D PAUSE-1      HALT                     Se asteaptă o întrerupere
                                           mascabilă.
                  DEC   BC                 Se decrementează contorul.
                  LD    A,B                Dacă contorul este astfel
                  OR    C                  redus la zero PAUSE a ajuns
                  JR    Z,1F4F,PAUSE-END   la sfîrsit.
                  LD    A,B                Dacă operandul a fost zero,
                  AND   C                  BC va contine acum +FFFF si
                  INC   A                  acestă valoare va fi readusă
                  JR    NZ,1F49,PAUSE-2    la zero. Salt pentru toate
                  INC   BC                 celelalte valori.
1F49 PAUSE-2      BIT   5,(FLAGS)          Salt înapoi numai dacă a fost
                  JR    Z,1F3D,PAUSE-1     apăsată o tastă.
```

Perioada pentru PAUSE s-a terminat acum.

```
1F4F PAUSE-END    RES   5,(FLAGS)          Semnalizare 'nici o tastă
                                           apăsată'.
                  RET                      Acum se revine - la STMT-RET.
```

THE 'BREAK-KEY' SUBROUTINE (SUBRUTINA 'TASTA BREAK')
Această subrutină este apelată în mai multe situatii pentru a citi tasta BREAK. Fanionul de transport este returnat resetat numai dacă tastele SHIFT si

BREAK au fost apăsate amîndouă deodată.

```
1F54 BREAK-KEY    LD    A,+7F          Se formează adresa portului
                  IN    A,(+FE)        +7FFE şi se citeşte într-un
                                       octet.
                  RRA                  Se examinează bitul 0 prin
                                       deplasare în poziţia
                                       transportului.
                  RET   C              Revenire dacă tasta BREAK nu
                                       a fost apăsată.
                  LD    A,+FE          Se formează adresa portului
                  IN    A,(+FE)        +FEFE şi se citeşte într-un
                                       octet.
                  RRA                  Se examinează din nou bitul
                                       0.
                  RET                  Revenire cu transportul
                                       resetat dacă ambele taste au
                                       fost apăsate.
```

THE 'DEF IN' COMMAND ROUTINE (RUTINA DE COMANDA 'DEF IN')
In timpul cît se verifică sintaxa, se verifică instrucţia DEF FN pentru a fi sigur că are forma corectă. Spaţiul este de asemenea făcut accesibil pentru rezultatul evaluării funcţiei.
In 'timpul execuţiei' instrucţiunea DEF FN este omisă.

```
1F60 DEF-FN       CALL  2530,SYNTAX-Z  Salt înainte dacă se verifică
                  JR    Z,1F6A,DEF-FN-1 sintaxa.
                  LD    A,+CE          Altfel se omite specificaţia
                  JP    1E39,PASS-BY   'DEF FN'.
```

Întîi se consideră variabila funcţiei.

```
1F6A DEF-FN-1     SET   6,(FLAGS)      Semnalizare o 'variabilă
                                       numerică'.
                  CALL  2C8D,ALPHA     Se verifică dacă prezentul
                                       cod este o literă.
                  JR    NC,1F89,DEF-FN-4 Salt înainte dacă nu este.
                  RST   0020,NEXT-CHAR Se aduce următorul caracter.
                  CP    +24            Salt înainte dacă nu este un
                  JR    NZ,1F7D,DEF-FN-2 '$'.
                  RES   6,(FLAGS)      Se schimbă bitul 6 ca şi cum
                                       ar fi o variabilă şir.
                  RST   0020,NEXT-CHAR Se aduce următorul caracter.
1F7D DEF-FN-2     CP    +28            '(' trebuie să urmeze numele
                  JR    NZ,1FBD,DEF-FN-7 variabilei.
                  RST   0020,NEXT-CHAR Se aduce următorul caracter.
                  CP    +29            Salt înainte dacă este ')' ca
                  JR    Z,1FA6,DEF-FN-6 şi cum nu există parametrii
                                       ai funcţiei.
```

Acum se introduce o buclă pentru lucrul cu fiecare parametru pe rînd.

```
1F86 DEF-FN-3     CALL  2C8D,ALPHA     Prezentul cod trebuie să fie
1F89 DEF-FN-4     JP    NC,1C8A,REPORT-C o literă.
                  EX    DE,HL          Se salvează indicatorul în
                                       DE.
                  RST   0020,NEXT-CHAR Se aduce următorul caracter.
                  CP    +24            Salt înainte dacă nu este un
                  JR    NZ,1F94,DEF-FN-5 '$'.
                  EX    DE,HL          Altfel se salvează imediat în
                                       DE noul indicator.
                  RST   0020,NEXT-CHAR Se aduce următorul caracter.
1F94 DEF-FN-5     EX    DE,HL          Se mută indicatorul ultimului
                                       caracter al numelui în
                                       registrul pereche HL.
                  LD    BC,+0006       Acum se fac sase locaţii după
                  CALL  1655,MAKE-ROOM ultimul caracter şi se
                  INC   HL             introduce un 'marcator de
                  INC   HL             număr' în prima dintre noile
                  LD    (HL),+0E       locaţii.
                  CP    +2C            Dacă prezentul caracter este
                  JR    NZ,1FA6,DEF-FN-6 '.', atunci salt înapoi ca şi
                  RST   0020,NEXT-CHAR cum ar trebui să mai fie un
                  JR    1F86,DEF-FN-3  parametru în continuare;
                                       altfel se sare afară din
                                       buclă.
```

In continuare se consideră definiţia funcţiei.

```
1FA6 DEF-FN-6     CP    +29              Se verifică faptul că ')'
                  JR    NZ,1FBD,DEF-FN-7 există.
                  RST   0020,NEXT-CHAR   Este adus noul caracter.
                  CP    +3D              El trebuie să fie '='.
                  JR    NZ,1FBD,DEF-FN-7
                  RST   0020,NEXT-CHAR   Se aduce următorul caracter.
                  LD    A,(FLAGS)        Se salvează natura variabilei
                  PUSH  AF               - numeric sau sir.
                  CALL  24FB,SCANNING    Acum se consideră definirea
                                         ca o expresie.
                  POP   AF               Se aduce natura variabilei si
                  XOR   (FLAGS)          se verifică dacă este de
                  AND   +40              acelasi tip cum s-a găsit
                                         pentru definitie.
1FBD DEF-FN-7     JP    NZ,1C8A,REPORT-C Se dă prezentarea erorii,
                                         dacă se cere.
                  CALL  1BEE,CHECK-END   Iesire prin subrutina
                                         CHECK-END. (Astfel se face
                                         transferul pentru a lua în
                                         considerare următoarea
                                         instructie de pe liniei.)
```

THE 'UNSTACK-Z' SUBROUTINE (SUBRUTINA 'UNSTACK-Z')
Această subrutină este apelată în cîteva cazuri pentru 'a se reveni mai devreme' dintr-o subrutină pentru verificarea sintaxei. Justificarea acestui fapt este ca să permită tipărirea caracterelor actuale sau trecrea valorii în/din stiva calculatorului.

```
1FC3 UNSTACK-Z    CALL  2530,SYNTAX-Z    A fost verificată sintaxa?
                  POP   HL               Se aduce adresa de revenire
                  RET   Z                dar se ignoră 'timpul
                                         sintaxă'.
                  JP    (HL)             In 'timpul executiei' se face
                                         o revenire simplă la rutina
                                         apelantă.
```

THE 'LPRINT & PRINT' COMMAND ROUTINES (RUTINELE DE COMANDA 'LPRINT & PRINT')
Canalul corespunzător este deschis după cum este necesar si informatia ce trebuie tipărită este considerată pe rînd.

```
1FC9 LPRINT       LD    A,+03            Pregătire pentru deschiderea
                                         canalului 'P'.
                  JR    1FCF,PRINT-1     Salt înainte.
1FCD PRINT        LD    A,+02            Pregătire pentru deschiderea
                                         canalului 'S'.
1FCF PRINT-1      CALL  2530,SYNTAX-Z    Un canal este deschis numai
                  CALL  NZ,1601,CHAN-OPEN dacă s-a verificat sintaxa.
                  CALL  0D4D,TEMPS       Setare temporară a variabilei
                                         sistem culoare.
                  CALL  1FDF,PRINT-2     Se apelează subrutina pentru
                                         controlul tipăririi.
                  CALL  1BEE,CHECK-END   Se mută pentru a considera
                  RET                    următoarea instructie; se
                                         face prin CHECK-END dacă s-a
                                         verificat sintaxa.
```

Subrutina pentru controlul tipăririi este apelată de rutinele de comandă PRINT, LPRINT si INPUT.

```
1FDF PRINT-2      RST   0018,GET-CHAR     Se aduce primul caracter.
                  CALL  2045,PR-END-Z    Salt în față dacă s-a ajuns
                  JR    Z,1FF2,PRINT-4   deja la sfîrsitul listei de
                                         informatii.
```

Acum se introduce o buclă pentru lucrul cu 'controlorii de poziti' si se tipăreste informatia.

```
1FE5 PRINT-3      CALL  204E,PR-POSN-1   Se lucrează cu orice
                  JR    Z,1FE5,PRINT-3   controlori de pozitie
                                         consecutivi.
                  CALL  1FFC,PR-ITEM-1   Se lucrează cu un singur
                                         număr de tipărire.
                  CALL  204E,PR-POSN-1   Se verifică următorii
                  JR    Z,1FE5,PRINT-3   controlori de pozitie si
                                         informatia de tipărit pînă
                                         cînd nu mai este nici unul.
1FF2 PRINT-4      CP    +29              Revenire dacă prezentul
                  RET   Z                caracter este ')'; altfel se
                                         consideră executia unui
```

'carriage return'.

THE 'PRINT A CARRIAGE RETURN' SUBROUTINE (SUBRUTINA 'TIPARIREA UUI CARRIAGE RETURN')

```
1FF5 PRINT-CR    CALL  1FC3,UNSTACK-Z      Revenire dacă se verifică
                                           sintaxa.
                 LD    A,+0D               Se tipăreste un caracter
                 RST   0010,PRINT-A-1      carriage return si se revine.
                 RET
```

THE 'PRINT ITEMS' SUBROUTINE (SUBRUTINA 'TIPARIRE INFORMATIE')
Această subrutină este apelată de rutinele de comandă PRINT, LPRINT si INPUT.
Tipurile diferite de tipărire a informatiei sînt identificate si tipărite.

```
1FFC PR-ITEM-1   RST   0018,GET-CHAR       Este adus primul caracter.
                 CP    +AC                 Salt mai departe numai dacă
                 JR    NZ,200E,PR-ITEM-2   este un 'AT'.
```

Acum se lucrează cu 'AT'.

```
                 CALL  1C79,NEXT-2NUM      Cei doi parametrii sînt
                                           transferati în stiva
                                           calculatorului.
                 CALL  1FC3,UNSTACK-Z      Revenire dacă se verifică
                                           sintaxa.
                 CALL  2307,STK-TO-BC      Parametrii sînt comprimati în
                                           registrul pereche BC.
                 LD    A,+16               Registrul A este încărcat cu
                 JR    201E,PR-AT-TAB      caracterul de control AT
                                           înainte de a se executa
                                           saltul.
```

Urmează căutarea unui 'TAB'.

```
200E PR-ITEM-2   CP    +AD                 Salt înainte numai dacă este
                 JR    NZ,2024,PR-ITEM-3   un 'TAB'.
```

Acum se lucrează cu 'TAB'.

```
                 RST   0020,NEXT-CHAR      Se aduce următorul caracter.
                 CALL  1C82,EXPT-1NUM      Se transferă un parametru în
                                           stiva calculatorului.
                 CALL  1FC3,UNSTACK-Z      Revenire dacă se verifică
                                           sintaxa.
                 CALL  1E99,FIND-INT2      Valoarea este comprimată în
                                           registrul pereche BC.
                 LD    A,+17               Registrul A este încărcat cu
                                           caracterul de control TAB.
```

Informatiile de tipărit 'AT' si 'TAB' sînt tipărite prin trei apelări PRINT-OUT.

```
201E PR-AT-TAB   RST   0010,PRINT-A-1      Se tipăreste caracterul de
                                           control.
                 LD    A,C                 Este urmat de prima valoare.
                 RST   0010,PRINT-A-1
                 LD    A,B
                 RST   0010,PRINT-A-1      In final se tipăreste a doua
                 RET                       valoare; apoi se revine.
```

Se consideră apoi introducerea informatiei de culoare.

```
2024 PR-ITEM-3   CALL  21F2,CO-TEMP-3      Revenire cu transportul
                                           resetat dacă s-a găsit o
                                           informatie de culoare.
                 RET   NC                  Dacă nu s-a găsit nici una,
                                           se continuă.
                 CALL  2070,STR-ALTER      Apoi se consideră dacă
                                           trebuie schimbat sirul.
                 RET   NC                  Se continuă numai dacă nu a
                                           fost alterat.
```

Acum informatia de tipărire trebuie să fie o expresie, fie numerică fie sir.

```
                 CALL  24FB,SCANNING       Se evaluează expresia, dar se
                 CALL  1FC3,UNSTACK-Z      revine dacă se verifică
                                           sintaxa.
                 BIT   6,(FLAGS)           Se testează natura expresiei.
```

```
                CALL  Z,2BF1,STK-FETCH      Dacă este sir, se aduc
                                            parametrii necesari; dar dacă
                JP    NZ,2DE3,PRINT-FP      este numerică, atunci se iese
                                            prin PRINT-FP.
```

Acum se instalează o buclă care să lucreze cu fiecare caracter al sirului pe rînd.

```
203C PR-STRING  LD    A,B                   Se revine dacă nu mai este
                OR    C                     nici un caracter în sir;
                DEC   BC                    altfel contorul este
                RET   Z                     decrementat.
                LD    A,(DE)                Se aduce codul si se
                INC   DE                    incrementează indicatorul.
                RST   0010,PRINT-A-1        Codul este tipărit si se face
                JR    203C,PR-STRING        un salt pentru a considera
                                            caracterele în continuare.
```

THE 'END OF PRINTING' SUBROUTINE (SUBRUTINA 'SFIRSITUL TIPARIRII')
Dacă nu mai este nimic de tipărit, fanionul zero va fi setat.

```
2045 PR-END-Z   CP    +29                   Revenire dacă este un
                RET   Z                     caracter ')'.
2048  PR-ST-END CP    +0D                   Acum se revine dacă este un
                RET   Z                     'carriage return'.
                CP    +3A                   Inainte de revenire se
                RET                         compară cu ':'.
```

THE 'PRINT POSITION' SUBROUTINE (SUBRUTINA 'POZITIE TIPARIRE')
Prin această subrutină se iau în considerare variatele pozitii ale caracterelor de control.

```
204E PR-POSN-1  RST   0018,GET-CHAR         Se aduce prezentul caracter.
                CP    +3B                   Salt înainte dacă este ';'.
                JR    Z,2067,PR-POSN-3
                CP    +2C
                JR    NZ,2061,PR-POSN-2     De asemenea se sare înainte
                CALL  2530,SYNTAX-Z         dacă este orice caracter
                JR    Z,2067,PR-POSN-3      diferit de ',' ; dar nu se
                                            tipăreste caracterul dacă se
                                            verifică sintaxa.
                LD    A,+06                 Se încarcă registrul A cu
                RST   0010,PRINT-A-1        codul de control 'virgulă' si
                JR    2067,PR-POSN-3        se tipăreste; apoi salt
                                            înainte;
2061 PR-POSN-2  CP    +27                   Este un '''?
                RET   NZ                    Revenire dacă nu este nici
                                            unul din controlorii de
                                            pozitie.
                CALL  1FF5,PR-CR            Se tipăreste un 'carriage
                                            return' numai dacă se
                                            verifică sintaxa.
2067 PR-POSN-3  RST   0020,NEXT-CHAR        Se aduce următorul caracter.
                CALL  2045,PR-END-Z         Dacă nu s-a ajuns la finalul
                JR    NZ,206E,PR-POSN-4     instructiunii de tipărire,
                POP   BC                    atunci se sare înainte;altfel
206E PR-POSN-4  CP    A                     se revine la rutina apelantă.
                RET                         Fanionul zero va fi resetat
                                            dacă nu s-a atins sfîrsitul
                                            instructiunii de tipărire.
```

THE 'ALTER STREAM' SUBROUTINE (SUBRUTINA 'SIR ALTERAT')
Această subrutină este apelată întotdeauna cînd este necesar să se considere dacă utlizatorul doreste să folosească un sir diferit.

```
2070 STR-ALTER  CP    +23                   Se revine cu fanionul de
                SCF                         transport setat numai dacă
                RET   NZ                    prezentul caracter este '#'.
                RST   0020,NEXT-CHAR        Avans CH-ADD.
                CALL  1C82,EXPT-1NUM        Se trece parametrul în stiva
                                            calculatorului.
                AND   A                     Se sterge fanionul de
                                            transport.
                CALL  1FC3,UNSTACK-Z        Revenire dacă se verifică
                                            sintaxa.
                CALL  1E94,FIND-INT1        Valoarea este trecută în
                                            registrul A.
                CP    +10                   Se dă prezentarea 0 dacă
                JP    NC,160E,REPORT-O      valoarea este peste +FF.
                CALL  1601,CHAN-OPEN        Se utilizează canalul pentru
```

```
                    AND   A                    sirul în discutie.
                    RET                        Se sterge fanionul de
                                               transport si se revine.
```

THE 'INPUT' COMMAND ROUTINE (RUTINA DE COMANDA 'INPUT')
Această rutină permite valorilor introduse de la tastatură să fie atribuite variabilelor. De asemenea este posibil să fie informatii de tipărire introduse în instructiunea INPUT si aceste informatii sînt tipărite în partea inferioară a ecranului.

```
2089 INPUT          CALL  2530,SYNTAX-Z        Salt înainte dacă s-a
                    JR    Z,2096,INPUT-1       verificat sintaxa.
                    LD    A,+01                Se deschide canalul 'K'.
                    CALL  1601,CHAN-OPEN
                    CALL  0D6E,CLS-LOWER       Se sterge partea inferioară a
                                               ecranului.
2096 INPUT-1        LD    (DF-SZ),+01          Se setează partea inferioară
                                               astfel încît să aibe mărimea
                                               de o linie.
                    CALL  20C1,IN-ITEM-1       Se apelează rutina pentru
                                               lucrul cu informatii INPUT.
                    CALL  1BEE,CHECK-END       Se trece la următoarea
                                               instructiunea dacă s-a
                                               verificat sintaxa.
                    LD    BC,(S-POSN)          Se aduce pozitia curentă
                                               pentru tipărit.
                    LD    A,(DF-SZ)            Salt înainte dacă pozitia
                    CP    B                    curentă este deasupra părtii
                    JR    C,20AD,INPUT-2       de jos a ecranului.
                    LD    C,+21                Altfel se setează pozitia de
                    LD    B,A                  tipărit în vîrful partii de
                                               jos a ecranului.
20AD INPUT-2        LD    (S-POSN),BC          Resetare S-POSN.
                    LD    A,+19                Acum se setează contorul de
                                               defilare.
                    SUB   B
                    LD    (SCR-RT),A
                    RES   0,(TV-FLAG)          Semnalizare 'ecran principal'
                    CALL  0DD9,CL-SET          Se setează variabilele sistem
                    JP    0D6E,CLS-LOWER       si se iese prin CLS-LOWER.
```

Informatiile INPUT si informatiile PRINT introduse sînt prelucrate pe rînd de bucla următoare.

```
20C1 IN-ITEM-1      CALL  204E,PR-POSN-1       Se consideră mai întîi orice
                    JR    Z,20C1,IN-ITEM-1     caractere de control al
                                               pozitiei.
                    CP    +28                  Salt înainte dacă prezentul
                    JR    NZ,20D8,IN-ITEM-2    caracter nu este '('.
                    RST   0020,NEXT-CHAR       Se aduce următorul caracter.
                    CALL  1FDF,PRINT-2         Acum se apelează rutina de
                                               comandă PRINT pentru tratarea
                                               informatiilor dintre
                                               paranteze.
                    RST   0018,GET-CHAR        Se aduce prezentul caracter.
                    CP    +29                  Se dă prezentarea C numai
                    JP    NZ,1C8A,REPORT-C     dacă este un caracter ')'.
                    RST   0020,NEXT-CHAR       Se aduce următorul caracter
                    JP    21B2,IN-NEXT-2       si salt înainte pentru a
                                               verifica dacă mai sînt în
                                               continuare informatii INPUT.
```

Acum se ia în considerare dacă s-a folosit INPUT LINE.

```
20D8 IN-ITEM-2      CP    +CA                  Salt înainte dacă nu este
                    JR    NZ,20ED,IN-ITEM-3    'LINE'.
                    RST   0020,NEXT-CHAR       Avans CH-ADD.
                    CALL  1C1F,CLASS-01        Se determină adresa de
                                               destinatie a variabilei.
                    SET   7,(FLAGX)            Semnalizare 'utilizare INPUT
                                               LINE'.
                    BIT   6,(FLAGS)            Se dă prezentarea C numai
                    JP    NZ,1C8A,REPORT-C     dacă se foloseste o variabilă
                                               sir.
                    JR    20FA,IN-PROMPT       Salt înainte pentru iesirea
                                               mesajului imediat.
```

Se trece la tratarea variabilelor simple INPUT.

```
20ED IN-IETM-3    CALL  2C8D,ALPHA          Salt înainte pentru ciclarea
                  JP    NC,21AF,IN-NEXT-1   buclei dacă prezentul
                                            caracter nu este o literă.
                  CALL  1C1F,CLASS-01       Se determină adresa de
                                            destinatie pentru variabilă.
                  RES   7,(FLAGX)           Semnalare 'nu este INPUT
                                            LINE'.
```

Mesajul imediat este acum construit în spatiul de lucru.

```
20FA IN-PROMPT    CALL  2530,SYNTAX-Z       Salt înainte numai dacă este
                  JP    Z,2182,IN-NEXT-2    verificată sintaxa.
                  CALL  16BF,SET-WORK       Spatiul de lucru este setat
                                            pe zero.
                  LD    HL,+5C71            Adresa FLAGX.
                  RES   6,(HL)              Semnalizare 'sir rezultat'.
                  SET   5,(HL)              Semnalizare 'mod INPUT'.
                  LD    BC,+0001            Se permite doar o locatie
                                            pentru mesajul imediat.
                  BIT   7,(HL)              Salt înainte dacă este
                                            folosit 'LINE'.
                  JR    NZ,211C,IN-PR-2
                  LD    A,(FLAGS)           Salt înainte dacă se asteaptă
                  AND   +40                 o intrare numerică.
                  JR    NZ,211A,IN-PR-1
                  LD    C,+03               O intrare sir va necesita
                                            trei locatii.
211A IN-PR-1      OR    (HL)                Pentru o intrare numerică
                  LD    (HL),A              bitul 6 al FLAGX va fi setat.
211C IN-PR-2      RST   0030,BC-SPACES      Este făcut accesibil numărul
                                            de locatii cerut.
                  LD    (HL),+0D            In ultima locatie se trece un
                                            'carriage return'.
                  LD    A,C                 Se testează bitul 6 din
                  RRCA                      registrul C si salt înainte
                  RRCA                      dacă este cerută doar o
                  JR    NC,2129,IN-PR-3     locatie.
                  LD    A,+22               Caracterul 'limite duble'
                  LD    (DE),A              este încărcat în prima si în
                  DEC   HL                  cea de a doua locatie.
                  LD    (HL),A
2129 IN-PR-3      LD    (K-CUR),HL          Acum se poate salva pozitia
                                            cursorului.
```

In cazul INPUT LINE editorul poate fi apelat fără pregătiri, dar pentru alte itpuri de INPUT stiva de eroare trebuie schimbată astfel încît să prindă erorile.

```
                  BIT   7,(FLAGX)           Salt înainte cu 'INPUT LINE'.
                  JR    NZ,215E,IN-VAR-3
                  LD    HL,(CH-ADD)         Se salvează valoarea curentă
                  PUSH  HL                  a lui CH-ADD & ERR-SP în
                  LD    HL,(ERR-SP)         stiva calculatorului.
                  PUSH  HL
213A IN-VAR-1     LD    HL,+213A            Acesta va fi 'punctul de
                  PUSH  HL                  revenire' în caz de eroare.
                  BIT   4,(FLAGS2)          Se schimbă doar indicatorul
                  JR    Z,2148,IN-VAR-2     stivei de eroare dacă se
                  LD    (ERR-SP),SP         foloseste canalul 'K'.
2148 IN-VAR-2     LD    HL,(WORKSP)         Setare HL la începutul liniei
                  CALL  11A7,REMOVE-FP      INPUT si se elimină orice
                                            forme de punct flotant. (Nu
                                            se face nici o exceptie,
                                            poate doar după o eroare.)
                  LD    (ERR-NR),+FF        Semnalizare 'încă nu este
                                            eroare'.
                  CALL  0F2C,EDITOR         Acum se ia INPUT si cu
                  RES   7,(FLAGS)           fanionul sintaxă/executie
                  CALL  2189,IN-ASSIGN      indicînd sintaxă, se verifică
                  JR    2161,IN-VAR-4       INPUT pentru erori; salt dacă
                                            este în ordine; dacă nu, se
                                            revine în IN-VAR-1.
215E IN-VAR-3     CALL  0F2CC,EDITOR        Se aduce 'LINE'.
```

Toate variabilele sistem trebuie resetate înainte de a se putea face actuala atribuire.

```
2161 IN-VAR-4     LD    (K-CUR-hi),+00      Adresa cursorului este
                                            resetată.
```

```
                CALL  21D6,IN-CHAN-K      Se execută un salt dacă se
                JR    NZ,2174,IN-VAR-5    foloseste alt canal decît
                                          canalul 'K'.
                CALL  111D,ED-COPY        Linia intrare este copiată pe
                LD    BC,(ECHO-E)         ecran si pozitia din ECHO-E
                CALL  0DD9,CL-SET         dă pozitia curentă din partea
                                          de jos a ecranului.
2174 IN-VARS-5  LD    HL,+5C71            Adresa FLAGX.
                RES   5,(HL)              Semnalizare 'mod editare'.
                BIT   7,(HL)              Salt înainte dacă se tratează
                RES   7,(HL)              INPUT LINE.
                JR    NZ,219B,IN-VAR-6
                POP   HL                  Se coboară adresa IN-VAR-1.
                POP   HL                  Se restează ERR-SP la adresa
                LD    (ERR-SP),HL         ei originară.
                POP   HL                  Se salvează adresa originară
                LD    (X-PTR),HL          a lui CH-ADD în X-PTR.
                SET   7,(FLAGS)           Acum, cu fanionul 'sintaxă/
                CALL  2189,IN-ASSIGN      executie' indicînd 'executie'
                                          se face atribuirea.
                LD    HL,(X-PTR)          Se reface adresa originară a
                LD    (X-PTR-hi),+00      lui CH-ADD si se sterge X-PTR
                LD    (CH-ADD),HL
                JR    21B2,IN-NEXT-2      Salt înainte pentru a vedea
                                          dacă mai sînt în continuare
                                          informatii INPUT.
219B IN-VARS-6  LD    HL,(STKBOT)         S-a găsit lungimea lui 'LINE'
                LD    DE,(WORKSP)         în spatiul de lucru.
                SCF
                SBC   HL,DE
                LD    B,H                 DE indică începutul iar BC
                LD    C,L                 contine lungimea.
                CALL  2AB2,STK-ST-$       Acesti parametrii sînt
                CALL  2AFF,LET            stocati si se face actuala
                                          atribuire.
                JR    21B2,IN-NEXT-2      Salt înainte pentru a lua în
                                          considerare următoarele
                                          informatii.
```

Se consideră următoarele informatii din instructiunea INPUT.

```
21AF IN-NEXT-1  CALL  1FFC,PR-ITEM-1      Se tratează orice informatie
                                          de tipărit.
21B2 IN-NEXT-2  CALL  204E,PR-POSN-1      Se tratează orice controlori
                                          de pozitie.
                JP    Z,20C1,IN-ITEM-1    Se ciclează din nou bucla
                RET                       dacă mai sînt informatii în
                                          continuare; altfel revenire.
```

THE 'IN-ASSIGN' SUBROUTINE (SUBRUTINA 'IN-ASSIGN')
Această subrutină este apelată de două ori pentru fiecare valoare INPUT. O dată cu fanionul sintaxă/executie resetat (sintaxă) si o dată setat (executie).

```
2189 IN-ASSIGN  LD    HL,(WORKSP)         Se setează CH-ADD pentru a
                LD    (CH-ADD),HL         indica prima locatie a
                RST   0018,GET-CHAR       spatiului de lucru si se
                                          aduce caracterul.
                CP    +E2                 Este un caracter 'STOP'?
                JR    Z,21D0,IN-STOP      Salt dacă este.
                LD    A,(FLAGX)           Altfel se face atribuirea
                CALL  1C59,VAL-FET-2      'valorii' la variabilă.
                RST   0018,GET-CHAR       Se aduce caracterul prezent
                CP    +0D                 si se verifică dacă este un
                RET   Z                   'carriage return'. Revenire
                                          dacă este.
```

Prezentarea C - Nonsense în BASIC

```
21CE REPORT-C   RST   0008,ERROR-1        Se apelează rutina de tratare
                DEFB  +0B                 eroare.
```

Aici se vine dacă linia INPUT începe cu 'STOP'.

```
21D0 IN-STOP    CALL  2530,SYNTAX-Z       Nu se dă prezentarea eroare
                RET   Z                   trecerea sintaxei.
```

Prezentarea H - STOP în INPUT

```
21D4 REPORT-H     RST   0008,ERROR-1        Se apelează rutina de tratare
                  DEFB  +10                 eroare.
```

THE 'IN-CHAN-K' SUBROUTINE (SUBRUTINA 'IN-CHAN-K')
Această subrutină revine cu fanionul zero resetat numai dacă s-a folosit canalul 'K'.

```
21D6 IN-CHAN-K    LD    HL,(CURCHL)         Adresa de bază a canalului de
                  INC   HL                  informatie este adusă si
                  INC   HL                  codul canalului este comparat
                  INC   HL                  cu caracterul 'K'.
                  INC   HL
                  LD    A,(HL)
                  CP    +4B
                  RET                       Apoi se revine.
```

THE 'COLOUR ITEM' ROUTINES (RUTINELE 'INFORMATII CULOARE')
Acest set de rutine poate fi împărtit în două părti:

i. Introducerea tratării 'informatiei culoare'.
ii. Tratarea 'variabilei sistem culoare'.

i. Introducerea informatiei culoare este tratată prin apelarea subrutinei PRINT-OUT, după cum se cere.
Se introduce o buclă pentru tratarea fiecăarei informatii pe rînd. Punctul de intrare este CO-TEMP-2.

```
21E1 CO-TEMP-1      RST   0020,NEXT-CHAR        Se consideră următorul
                                                caracter în instructia BASIC.
21E2 CO-TEMP-2      CALL  21F2,CO-TEMP-3        Salt înainte pentru a vedea
                                                dacă prezentul cod reprezintă
                                                o introducere temporară a
                                                informatiei culoare.
                    RET   C                     Se revine cu transportul
                                                setat dacă nu este o
                                                informatie culoare.
                    RST   0018,GET-CHAR         Se aduce prezentul caracter.
                    CP    +2C                   Salt înapoi dacă este fie o
                    JR    Z,21E1,CO-TEMP-1      ',' fie un ';'; altfel a fost
                    CP    +3B                   o eroare.
                    JR    Z,21E1,CO-TEMP-1
                    JP    1C8A,REPORT-C         Iesire prin 'prezentarea C'.
21F2 CO-TEMP-3      CP    +D9                   Revenire cu fanionul de
                                                transport.
                    RET   C                     Se setează dacă codul nu este
                    CP    +DF                   între limitele +D9 la +DE
                                                (INK la OVER).
                    CCF
                    RET   C
                    PUSH  AF                    Codul informatiei de culoare
                    RST   0020,NEXT-CHAR        este păstrat pînă cînd CH-ADD
                    POP   AF                    a avansat la adresa
                                                parametrului care îi urmează.
```

Codul informatiei culoare si parametrul sînt acum 'tipăriti' apelînd PRINT-OUT în două ocazii.

```
21FC CO-TEMP-4      SUB   +C9                  Intervalul dat (+D9 la +DE)
                                                este redus la intervalul
                                                caracterului de control 8+10
                                                la +15).
                    PUSH  AF                    Codul caracterului de control
                    CALL  1C82,EXPT-1NUM        este păstrat pînă cînd
                    POP   AF                    parametrul este mutat în
                                                stiva calculatorului.
                    AND   A                     In acest punct se face o
                    CALL  1FC3,UNSTACK-Z        revenire dacă se verifică
                                                sintaxa.
                    PUSH  AF                    Codul caracterului de control
                    CALL  1E94,FIND-INT1        este păstrat pînă cînd
                    LD    D,A                   parametrul este mutat în
                    POP   AF                    registrul D.
                    RST   0010,PRINT-A-1        Caracterul de control este
                                                trimis afară.
                    LD    A,D                   Apoi este adus parametrul si
                    RST   0010,PRINT-A-1        trimis afară înainte de a se
                    RET                         reveni.
```

ii. Variabilele sistem culoare - ATTR-T, MASK-T & P-FLAG - sînt modificate după cum se cere. Această subrutină este apelată de PRINT-OUT. La intrare codul caracterului de control este în registrul A iar parametrul este în registrul D.
De notat că toate schimbările sînt pentru variabilele sistem 'temporare'.

```
2211 CO-TEMP-5      SUB   +11                  Se reduce ordinul si salt
                    ADC   A,+00                 înainte cu INK & PAPER.
                    JR    Z,2234,CO-TEMP-7
                    SUB   +02                   Se reduce încă o dată ordinul
                    ADC   A,+00                 si salt înainte cu FLASH &
                    JR    Z,2273,CO-TEMP-C      BRIGHT.
```

Codul de control al culorii va di acum +01 pentru INVERSE si +02 pentru OVER, iar variabila sistem P-FLAG este modificată corespunzător.

```
                    CP    +01                   Pregătire salt cu OVER.
```

```
                    LD    A,D          Se aduce parametrul.
                    LD    B,+01        Pregătire mască pentru OVER.
                    JR    NZ,2228,CO-TEMP-6  Acum salt.
                    RLCA               Bitul 2 din registrul A
                    RLCA               trebuie resetat pentru
                    LD    B,+04        INVERSE 0 si setat pentru
                                       INVERSE 1; masca va avea
                                       bitul 2 setat.
2228 CO-TEMP-6      LD    C,A          Registrul A este salvat cît
                                       timp este testat ordinul.
                    LD    A,D          Ordinul corect pentru INVERSE
                    CP    +02          si OVER este numai '0 - 1'.
                    JR    NC,2244,REPORT-K
                    LD    A,C          Se aduce registrul A.
                    LD    HL,+5C91     P-FLAG trebuie schimbat.
                    JR    226C,CO-CHANGE  Iesire prin CO-CHANGE si se
                                       modifică P-FLAG folosind 'B'
                                       ca mască. De fapt bit 0
                                       pentru OVER & bit 2 pentru
                                       INVERSE.
```

PAPER & INK sînt prelucrate de rutina următoare. La intrare fanionul de transport este setat pentru INK.

```
2234 CO-TEMP-7      LD    A,D          Se aduce parametrul.
                    LD    B,+07        Pregătire mască pentru INK.
                    JR    C,223E,CO-TEMP-8  Salt înainte cu INK.
                    RLCA               Se multiplică de opt ori
                    RLCA               parametrul pentru PAPER.
                    RLCA
                    LD    B,+38        Pregătire mască pentru PAPER.
223E CO-TEMP-8      LD    C,A          Se salvează parametrul în
                                       registrul C pînă cînd se
                                       testează rangul parametrului.
                    LD    A,D          Se aduce valoarea intială.
                    CP    +0A          Pentru PAPER/INK se permite
                    JR    C,2246,CO-TEMP-9  un ordin doar între '0 la 9'.
```

Prezentarea K - Culoare invalidă

```
2244 REPORT         RST   0008,ERROR-1  Se apelează rutina de tratare
                    DEFB  +13          eroare.
```

Se continuă tratarea PAPER & INK.

```
2246 CO-TEMP-9      LD    HL,+5C8F     Pregătire modificare ATTR-T.
                    CP    +08          Salt înainte cu PAPER/INK
                    JR    C,2258,CO-TEMP-B  '0 la 5'7'.
                    LD    A,(HL)       Se aduce valoarea curentă a
                    JR    Z,2257,CO-TEMP-A  lui ATTR-T si se foloseste
                                       neschimbată, sărind înainte,
                                       cu PAPER/INK '8'.
                    OR    B            Dar pentru PAPER/INK '9'
                    CPL                culorile pentru PAPER si INK
                    AND   +24          trebuie să fie negru si alb.
                    JR    Z,2257,CO-TEMP-A  Salt pentru INK/PAPER negru;
                    LD    A,B          se continuă pentru INK/PAPER
                                       alb.
2257 CO-TEMP-A      LD    C,A          Se mută valoarea în registrul
                                       C.
```

Masca (B) si valoarea (C) sînt acum folosite pentru a schimba ATTR-T.

```
2258 CO-TEMP-B      LD    A,C          Se mută valoarea.
                    CALL  226C,CO-CHANGE  Acum se schimbă ATTR-T, după
                                       cum este necesar.
```

In continuare se consideră MASK-T.

```
                    LD    A,+07        Bitii lui MASK-T sînt setati
                    CP    D            numai cînd se foloseste
                    SBC   A,A          PAPER/INK '8' sau '9'.
                    CALL  226C,CO-CHANGE  Acum se schimbă MASK-T, după
                                       cum este necesar.
```

Acum se consideră P-FLAG.

```
                    RLCA               Masca corespunzătoare este
                    RLCA               construită în registrul B si
```

```
                AND   +50          trebuie să schimbe bitii 4 &
                LD    B,A          6, după cum este necesar.
                LD    A,+08        Bitii lui P-FLAG sînt setati
                CP    D            doar cînd se foloseste PAPER/
                SBC   A,A          INK '9'.
                                   Se continuă în CO-CHANGE
                                   prelucrarea lui P-FLAG.
```

THE 'CO-CHANGE' SUBROUTINE (SUBRUTINA 'CO-CHANGE')
Această subrutină este folosită pentru 'a imprima' unei variabile sistem 'natura' bitilor registrului A. Registrul B contine o mască ce arată care biti trebuie 'copiati peste' din A în (HL).

```
226C CO-CHANGE  XOR   (HL)         Bitii, specificati de masca
                AND   B            din registrul B, au valoarea
                XOR   (HL)         schimbată si rezultatul trece
                LD    (HL),A       pentru a forma variabila
                                   sistem.
                INC   HL           Mutare la adresa următoarei
                                   variabile sistem.
                LD    A,B          Revenire cu masca în
                RET                registrul A.
```

Următoarea rutină tratează FLASH & BRIGHT.

```
2273 CO-TEMP-C  SBC   A,A          Fanionul zero este setat
                                   pentru BRIGHT.
                LD    A,D          Parametrul este încărcat si
                RRCA               rotit.
                LD    B,+80        Pregătire mască pentru FLASH.
                JR    NZ,227D,CO-TEMP-D  Salt înainte cu FLASH.
                RRCA               Rotire timp aditional si
                LD    B,+40        pregătire mască pentru BRIGHT
227D CO-TEMP-D  LD    C,A          Se salvează valoarea în
                                   registrul C.
                LD    A,D          Se aduce parametrul si i se
                CP    +08          testează ordinul; se permite
                JR    Z,2287,CO-TEMP-E  doar '0', '1' si '8'.
                CP    +02
                JR    NC,2244,REPORT-K
```

Acum se poate modifica variabila sistem ATTR-T.

```
2287 CO-TEMP-E  LD    A,C          Se aduce valoarea.
                LD    HL,+5C8F     Adresa ATTR-T.
                CALL  226C,CO-CHANGE  Acum se schimbă variabila
                                   sistem.
```

Acum se consideră valoarea din MASK-T.

```
                LD    A,C          Se aduce din nou valoarea.
                RRCA               Bitul setat din FLASH/BRIGHT
                RRCA               '8' (bitul 3) este mutat în
                RRCA               bitul 7 (pentru FLASH) sau
                                   în bitul 6 (pentru BRIGHT).
                JR    226C,CO-CHANGE  Iesire prin CO-CHANGE.
```

THE 'BORDER' COMMAND ROUTINE (RUTINA DE COMANDA 'BORDER')
Parametrul comenzii BORDER este folosit cu o comandă OUT pentru actuala schimbare a culorii marginii. Apoi parametrul este salvat în variabila sistem BORDER.

```
2294 BORDER     CALL  1E94,FIND-INT1  Parametrul este adus si i se
                CP    +08          testează rangul.
                JR    NC,2244,REPORT-K
                OUT   (+FE),A      Instructiunea OUT este apoi
                                   folosită pentru setarea
                                   culorii marginii.
                RLCA               Apoi parametrul este
                RLCA               multiplicat de opt ori.
                RLCA
                BIT   5,A          Dacă culoarea marginii este o
                JR    NZ,22A6,BORDER-1  culoare 'luminoasă' atunci
                                   culoarea INK (cernelii) din
                                   spatiul de editare trebuie
                                   să fie neagră - se execută
                                   un salt.
                XOR   +07          Se schimbă culoarea lui INK.
22A6 BORDER-1   LD    (BORDCR),A   Se setează variabila sistem,
```

```
                RET                          după cum se cere si se
                                             revine.
```

THE 'PIXEL ADDRESS' SUBROUTINE (SUBRUTINA 'ADRESA PIXEL')
Această subrutină este apelată de subrutina POINT si de rutina de comandă PLOT. Ea este introdusă cu coordonatele unui pixel în registrul pereche BC si revine cu registrul pereche HL continînd adresaoctetului fisierului de afisaj care contine acest pixel si cu A indicînd pozitia pixelului în cadrul octetului.

```
22AA PIXEL-ADD  LD    A,+AF                  Se testează dacă coordonata y
                SUB   B                      (în B) nu este mai mare decît
                JP    C,24F9,REPORT-B        175.
                LD    B,A                    Acum B contine 175 minus y.
                AND   A                      A contine b7b6b5b4b3b2b1b0,
                RRA                          bit din B. Si acum
                                             0b7b6b5b4b3b2b1.
                SCF
                RRA                          Acum 10b7b6b5b4b3b2.
                AND   A
                RRA                          Acum 010b7b6b5b4b3.
                XOR   B
                AND   +F8                    In final 010b7b6b2b1b0, asa
                XOR   B                      că H devine 64 + 8*INT(B/64)+
                LD    H,A                    B(mod 8), octetul superior al
                LD    A,C                    adresei pixelului. C contine
                                             X.
                RLCA                         A începe cu c7c6c5c4c3c2c1c0.
                RLCA
                RLCA                         Acum este c2c1c0c7c6c5c4c3.
                XOR   B
                AND   +C7
                XOR   B                      Acum c2c1b5b4b3c5c4c3.
                AND   +C7
                XOR   B
                RLCA
                RLCA                         In final b5b4b3c7c6c5c4c3,
                LD    L,A                    asa că L devine 32*INT (B(mod
                LD    A,C                    64)/8)+INT(x/8), bit inferior
                AND   +07                    A contine x(mod 8): asa că
                RET                          pixelul este bitul (A-7) în
                                             cadrul octetului.
```

THE 'POINT' SUBROUTINE (SUBRUTINA 'PUNCT')
Această subrutină este apelată de functia POINT în SCANNING. Este introdusă cu coordonatele pixelului în stiva calculatorului si readuce o ultimă valoare de 1 dacă pixelul este culoarea cernelii, si 0 dacă este culoarea hîrtiei.

```
22CB POINT-SUB  CALL  2307,STK-TO-BC         Coordonata Y în B, x în C.
                CALL  22AA,PIXEL-ADD         Adresa pixelului în HL.
                LD    B,A                    B va număra A+1 bucle pentru
                INC   B                      a primi bitul dorit din (HL)
                LD    A,(HL)                 în locatia 0.
22D4 POINT-LP   RLCA                         Deplasare.
                DJNZ  22D4,POINT-LP
                AND   +01                    Bitul este 1 pentru cerneală,
                                             0 pentru hîrtie.
                JP    2D28,STACK-A           Este pus în stiva
                                             calculatorului.
```

THE 'PLOT' COMMAND ROUTINE (RUTINA DE COMANDA 'PLOT')
Această rutină constă dintr-o subrutină principală plus o linie pentru apelarea ei si o linie pentru iesirea din ea. Rutina principală este utilizată de două ori de CIRCLE iar subrutina este apelată de DRAW. Rutina este introdusă avînd coordonatele unui pixel în stiva calculatorului. Aceasta găseste adresa acelui pixel si o marchează, tinînd cont de stările lui INVERSE si OVER continute în P-FLAG.

```
22DC PLOT       CALL  2307,STK-TO-BC         Coordonata Y în B, x în C.
                CALL  22E5,PLOT-SUB          Se apelează subrutina.
                JP    0D4D,TEMPS             Iesire, se setează temporar
                                             culoarea.
22E5 PLOT-SUB   LD    (COORDS),BC            Este setată variabila sistem.
                CALL  22AA,PIXEL-ADD         Adresa pixelului în HL.
                LD    B,A                    B va număra A+1 bucle pentru
                INC   B                      a pune un zero în locul
                                             potrivit din A.
                LD    A,+FE                  Se introduc zeroturi.
22F0 PLOT-LOOP  RRCA                         Apoi se alipeste bitul de
```

```
                    DJNZ  22F0,PLOT-LOOP      pozitie al pixelului în octet
                    LD    B,A                 Apoi se copiază în B.
                    LD    A,(HL)              Octetul pixelului este
                                              obtinut în A.
                    LD    C,(P-FLAG)          S-a obtinut P-FLAG si mai
                    BIT   0,C                 întîi se testează fată de
                                              OVER.
                    JR    NZ,22FD,PL-TST-IN   Salt dacă OVER este 1.
                    AND   B                   OVER 0 face întîi pixel zero.
22FD PL-TST-IN      BIT   2,C                 Se testează INVERSE.
                    JR    NZ,2303,PLOT-END    INVERSE 1 lasă pixelul cum a
                                              fost (OVER 1) sau zero (OVER
                                              0).
                    XOR   B                   INVERSE 0 lasă pixelul
                    CPL                       complementat (OVER 1) sau 1
                                              (OVER 0).
2303 PLOT-END       LD    (HL),A              Este introdus octetul.
                                              Ceilalti biti ai săi rămîn
                                              neschimbati în orice caz.
                    JP    0BDB,PO-ATTR        Iesire, setare octet atribut.
```

THE 'STK-TO-BC' SUBROUTINE (SUBRUTINA 'STK-TO-BC')
Această subrutină încarcă două numere în punct flotant în registrul pereche BC. Aceasta va alege parametrii din intervalul +00 - +FF. Ea obtine de asemenea în DE 'mutarea diagonală' a valorii (±1,±1) care este utilizată în subrutina de desenare linie DRAW.

```
2307 STK-TO-BC      CALL  2314,STK-TO-A       Primul număr în A.
                    LD    B,A                 Deci în B.
                    PUSH  BC                  Se salvează pentru scurt timp
                    CALL  2314,STK-TO-A       Al doilea număr în A.
                    LD    E,C                 Semnul lui indicator în E.
                    POP   BC                  Se reface primul număr.
                    LD    D,C                 Semnul lui indicator în D.
                    LD    C,A                 Al doilea număr în C.
                    RET                       Acum BC, DE sînt asa cum s-a
                                              cerut.
```

THE 'STK-TO-A' SUBROUTINE (SUBRUTINA 'STK-TO-A')
Această subrutină încarcă registrul A cu numărul în virgulă flotantă continut în vîrful stivei calculatorului. Numărul trebuie să fie în limitele 00-FF.

```
2314 TSK-TO-A       CALL  2DD5,FP-TO-A        Modulul ultimei valori
                    JP    C,24F9,REPORT-B     rotite în A, dacă este
                                              posibil; altfel se dă eroare.
                    LD    C,+01               Se pune unu în C pentru
                                              ultima valoare pozitivă.
                    RET   Z                   Revenire dacă valoarea a fost
                                              pozitivă.
                    LD    C,+FF               Altfel se schimbă în C +FF
                    RET                       (de fapt minus 1). Revenire.
```

THE CIRCLE COMMAND ROUTINE (RUTINA DE COMANDA CERC)
Această rutină reprezintă o aproximatie a cercului cu centrul de coordonate X si Y si de rază Z. Aceste numere se rotunjesc la cel mai apropiat întreg înainte de a se utiliza. Acest Z trebuie să fie mai mic decît 87,5, chiar dacă (X,Y) sînt în centrul ecranului. Metoda utilizată este de a reprezenta o serie de arcuri aproximate prin linii drepte. Ea este ilustrată într-un program BASIC dat în anexă. Notările din program sînt prezentate în continuare.

CIRCLE are patru părti:

i. Se testează raza. Dacă ea este în modul mai mică decît 1, se marchează doar X,Y;
ii. Se apelează CD-PRMS1 la 2470-24B6, care se utilizează pentru setarea parametrilor initiali atît pentru CIRCLE cît si pentru DRAW.;

iii. Se setează parametrii rămasi ai lui CIRCLE, incluzînd si înlocuirea initială pentru primul 'arc' (o linie dreaptă, de fapt);

iv. Se sare în DRAW pentru utilizarea buclei reprezentare-arc la 2420-24FA.

Acum se explică pe rînd părtile i. la iii.

i. 2320-23AA. Raza, numită Z', este obtinută din stiva calculatorului. Modulul ei Z este format si folosit de acum încolo. Dacă Z este mai mic decît 1, este sters din stivă si punctul X,Y este marcat printr-un salt în PLOT.

```
2320 CIRCLE         RST   0018,GET-CHAR       Se aduce prezentul caracter.
```

```
                CP    +2C                       Test fată de virgulă.
                JP    NZ,1C8A,REPORT-C          Dacă nu este virgulă, se dă
                                                eroare.
                RST   0020,NEXT-CHAR            Se aduce următorul caracter
                                                (raza).
                CALL  1C82,EXPT-1NUM            Raza în stiva calculatorului.
                CALL  1BEE,CHECK-END            Se trece la următoarea
                                                instructiune dacă se verifică
                                                sintaxa.
                RST   0028,FP-CALC              Se foloseste calculatorul:
                DEFB  +2A,abs                   stiva contine: X,Y,Z.
                DEFB  +3D,re-stack              Z este restocat; exponentul
                DEFB  +38,end-calc              lui este accesibil.
                LD    A,(HL)                    se aduce exponentul razei.
                CP    +81                       Se testează dacă raza este
                                                mai mică decît 1.
                JR    NC,233B,C-R-GRE-1         Dacă nu este, salt.
                RST   0028,FP-CALC              Dacă este mai mică, se sterge
                                                din stivă.
                DEFB  +02,delete                Stiva contine X,Y.
                DEFB  +38,end-calc
                JR    22DC,PLOT                 Se reprezintă punctul doar
                                                prin X,Y.
```

ii. 233B-2346 si apel CD-PRMS1. 2*PI este stocat în mem-5 si CD-PRMS1 este apelat. Această subrutină stochează în registrul B numărul de arce cerut pentru cer, si anume A=4*INT(PI*SQR Z/4)+4, deci 4,8,12..., pînă la maxim 32. De asemenea ea stochează în mem-0 la mem-4 cantitătile 2*PI/A, SIN(PI/A), 0, COS (2*PI/A) si SIN (2*PI/A).

```
233B C-R-GRE-1  RST   0028,FP-CLAC
                DEFB  +A3,stk-pi/2              X,Y,Z,PI/2
                DEFB  +38,end-calc              Acum se creste exponentul la
                LD    (HL),+83                  83 hex, schimbînd PI/2 în PI.
                RST   0028,FP-CALC              X,Y,Z,2*PI
                DEFB  +C5,st-mem-5              (2*PI este copiat în mem-5)
                DEFB  +02,delete                X,Y,Z
                DEFB  +38,end-calc
                CALL  247D,CD-PRMS1             Se setează parametrii intiali
```

iii. 2347-2381: parametrii rămasi si saltul la DRAW. Se face un test pentru a vedea dacă lungimea 'arcului' initial este mai mică decît 1. Dacă este, se face un salt simplu pentru reprezentarea X,Y. Altfel, parametrii sînt setati: X+Z si Y-Z*SIN (PI/A) sînt stocati de două ori ca punct de început si de sfîrsit, si copiati în COORDS; în mem-1 si mem-2 se stochează zero si 2*Z*SIN(PI/A) ca si cresteri initiale, dînd ca prim 'arc' linia verticală dreaptă împreună cu X+Z, Y-Z*SIN (PI/A) si X+Z, Y+Z*SIN (PI/A). Bucla trasare-arc din DRAW va asigura că toate punctele subsecventei rămîn în acelasi cerc ca aceste două puncte, cu unghiul incremental 2*PI/A. Dar este clar că aceste două puncte de fapt subîntind acest unghi în punctul X+Z*(1-COS (PI/A)),Y si nu în X,Y. Deci punctul de sfîrsit al fiecărui arc al cercului este înlocuit la dreapta de 2*(1-COS(PI/A)), care este mai mic decît o jumătate de pixel, si ajunge la cel mult un pixel.

```
2347            PUSH  BC                        Salvare contor-arc în B.
                RST   0028,FP-CALC              X,Y,Z
                DEFB  +31,duplicate             X,Y,Z,Z
                DEFB  +E1,get-mem-1             X,Y,Z,Z,SIN (PI/A)
                DEFB  +04,multiply              X,Y,Z,Z*SIN (PI/A)
                DEFB  +38,end-calc              SIN (PI/A) este jumătate din
                LD    A,(HL)                    lungimea 'arcului' initial;
                CP    +80                       este testat pentru a vedea
                                                dacă este mai mic de 0,5.
                JR    NC,235A,C-ARC-GE1         Dacă nu, se execută saltul.
                RST   0028,FP-CALC              Altfel, Z este sters din
                DEFB  +02,delete                stivă, de asemenea cu
                DEFB  +02,delete                jumătate de arc; stiva
                DEFB  +38,end-calc              calculatorului este stearsă;
                                                si salt pentru reprezentarea
                                                X,Y.
                POP   BC
                JP    22DC,PLOT
235A C-ARC-GE1  RST   0028,FP-CALC              X,Y,Z,Z*SIN (PI/A)
                DEFB  +C2,st-mem-2              (Z*SIN (PI/A) trece de acum
                                                în mem-2)
                DEFB  +01,exchange              X,Y,Z*SIN (PI/A),Z
                DEFB  +C0,st-mem-0              X,Y,Z*SIN (PI/A),Z
                DEFB  +02,delete                X,Y,Z*SIN (PI/A)
                DEFB  +03,subtract              X,Y-Z*SIN (PI/A)
```

```
         DEFB  +01,exchange          Y-Z*SIN (PI/A),X
         DEFB  +E0,get-mem-0         Y-Z*SIN (PI/A),X,Z
         DEFB  +0F,addition          Y-Z*SIN (PI/A),X+Z
         DEFB  +C0,st-mem-0          (X+Z este copiat în mem-0)
         DEFB  +01,exchange          X+Z,Y-Z*SIN (PI/A)
         DEFB  +31,duplicate         X+Z,Y-Z*SIN (PI/A),Y-Z*SIN
                                     (PI/A)
         DEFB  +E0,get-mem-0         sa,sb,sb,sa
         DEFB  +01,exchange          sa,sb,sa,sb
         DEFB  +31,duplicate         sa,sb,sa,sb,sb
         DEFB  +E0,get-mem-0         sa,sb,sa,sb,sb,sa
         DEFB  +A0,stk-zero          sa,sb,sa,sb,sa,sb,0
         DEFB  +C1,st-mem-1          (mem-1 este setat pe zero)
         DEFB  +02,delete            sa,sb,sa,sb,sb,sa
         DEFB  +38,end-calc
```

(Aici cu sa s-a notat X+Z si cu sb s-a notat Y-Z*SIN (PI/A)).

```
         INC   (mem-2-1st)           Incrementînd octetul exponent
                                     al mem-2 se setează mem-2 cu
                                     2*Z*SIN (PI/A).
         CALL  1E94,FIND-INT1        Ultima valoare X+Z este
         LD    L,A                   mutată din stivă în A si se
                                     copiază în L.
         PUSH  HL                    Salvare în HL.
         CALL  1E94,FIND-INT1        Y-Z*SIN (PI/A) trece din
         POP   HL                    stivă în A si este copiată în
                                     H.
         LD    H,A                   Acum HL contine punctul
                                     initial.
         LD    (COORDS),HL           El este copiat în COORDS.
         POP   BC                    Se reface contro-arc.
         JP    2420,DRW-STEPS        Se face un salt la DRAW.
```

(stiva contine acum X+Z,Y-Z*SIN (PI/A),Y-Z*SIN (PI/A),X+Z)

THE DRAW COMMAND ROUTINE (RUTINA DE COMANDA DESENARE)
Această rutină este introdusă cu coordonatele punctului XO,YO în COORDS. Dacă se dau numai doi parametrii X,Y cu comanda DRAW, se va desena o aproximare la o linie dreaptă din punctul XO,YO la XO+X,YO+Y. Dacă se dă un al treilea parametru G, se va desena o aproximare a unui arc de cerc de la XO,YO la XO+X,YO*y rotirea făcîndu-se în sens antiorar cu un unghi de G radiani.
Rutina are patru părti:

i. Se desenează doar o linie dacă se dau numai 2 parametrii sau dacă diametrul cercului implicat este mai mic decît 1;

ii. Se apelează CD-PRMS1 la 247D-24B6 pentru setarea primilor parametrii;

iii. Se aranjează parametrii rămasi, incluzînd înlocuirea initială a primului arc.

iv. Se introduce bucla desenare-arc si se desenează arcul ca o serie de arce mici aproximate prin linii drepte, apelînd subrutina de desenare-linie la 24B7-24FA, cum este necesar.

Două subrutine, CD-PRMS1 si DRAW-LINE, urmează rutina principală. Cele patru părti de deasupra rutinei principale vor fi tratate pe rînd.

i. Dacă sînt numai doi parametrii, se execută un salt la LINE-DRAW la 2477. O linie este de asemenea desenată dacă valoarea Z=(ABS X+ABSY)/
ABS SIN (G/2) este maică decît 1. Z face legătura între 1 si 1,5 ori diametrul cercului implicat. In această sectiune mem-0 este setat cu SIN (G/2), mem-1 cu Y, si mem-5 cu G.

```
2382 DRAW      RST   0018,GET-CHAR      Se aduce caracterul curent.
               CP    +2C               Dacă este virgulă,
               JR    Z,238D,DR-3-PRMS  atunci salt.
               CALL  1BEE,CHECK-END    Se trece la următoarea
                                       instructie dacă se verifică
                                       sintaxa.
               JP    2477,LINE-DRAW    Salt tocmai la desenare linie
238D DR-3-PRMS RST   0020,NEXT-CHAR     Se aduce următorul caracter
                                       (unghiul).
               CALL  1C82,EXPT-1NUM    Unghiul se trece în stiva
                                       calculatorului.
               CALL  1BEE,CHECK-END    Se trece la următoarea
                                       instructie dacă se verifică
                                       sintaxa.
```

```
                RST   0028,FP-CALC          X,Y,G sînt în stivă.
                DEFB  +C5,st-mem-5          (G este copiat în mem-5.)
                DEFB  +A2,stk-half          X,Y,G,0.5
                DEFB  +04,multiply          X,Y,G/2
                DEFB  +1F,sin               X,Y,SIN(G/2)
                DEFB  +31,duplicate         X,Y,SIN(G/2),SIN(G/2)
                DEFB  +30,not               X,Y,SIN(G/2),(0/1)
                DEFB  +30,not               X,Y,SIN(G/2),(1/0)
                DEFB  +00,jump-true         X,Y,SIN(G/2)
                DEFB  +06,to DR-SIN-NZ      (Dacă SIN(G/2)=0, atunci
                DEFB  +02,delete            G=2*N*PI desenează o linie
                                            dreaptă.)
                DEFB  +38,end-calc          X,Y
                JP    2477,LINE-DRAW        Linia XO,YO la XO+X,YO+Y
23A3 DR-SIN-NZ  DEFB  +C0,st-mem-0          (SIN G/2 este copiat în
                                            mem-0)
                DEFB  +02,delete            X,Y sînt acum în stivă.
                DEFB  +C1,st-mem-1          (Y este copiat în mem-1)
                DEFB  +02,delete            X
                DEFB  +31,duplicate         X,X
                DEFB  +2A,abs               X,X' (X' = ABS X)
                DEFB  +E1,get-mem-1         X,X',Y
                DEFB  +01,exchange          X,Y,X'
                DEFB  +E1,get-mem-1         X,Y,X',Y
                DEFB  +2A,abs               X,Y,X',Y' (Y' = ABS Y)
                DEFB  +0F,addition          X,Y,X'+Y'
                DEFB  +E0,get-mem-0         X,Y,X'+Y',SIN (G/2)
                DEFB  +05,division          X,Y,(X'+Y')/SIN (G/2) = Z'
                DEFB  +2A,abs               X,Y,Z (Z = ABS Z')
                DEFB  +e0,get-mem-0         X,Y,Z,SIN (G/2)
                DEFB  +01,exchange          X,Y,SIN(G/2),Z
                DEFB  +3D,re-stack          (Z este restocat pentru a
                DEFB  +38,end-calc          asigura că exponentul lui
                                            este accesibil)
                LD    A,(HL)                Se aduce exponentul lui Z.
                CP    +81                   Dacă Z este mai mare sau egal
                JR    NC,23C1,DR-PRMS       cu 1, atunci salt.
                RST   0028,FP-CALC          X,Y,SIN(G/2),Z
                DEFB  +02,delete            X,Y,SIN(G/2)
                DEFB  +02,delete            X,Y
                DEFB  +38,end-calc          se desenează linia de la
                JP    2477,LINE-DRAW        XO,YO la XO+X,YO+Y.
```

ii. Doar se apelează CD-PRMS1. Această subrutină salvează în registrul B numărul celui mai mic arc cerut pentru arcul complet, si anume A=4*INT (G'*SQR Z/8)+4, unde G' = mod G, sau 252 dacă expresia depăseste 252 (cum se poate întîmpla cu o coardă mare si un unghi mic). Asa că A este 4, 8, 12 ..., pînă la 252. De asemenea subrutina stochează în mem-0 la mem-4 cantitătile G/A, SIN (G/2*A), SIN (G/A).

```
23C1 DR-PRMS    CALL  247D,CD-PRMS1         Subrutina este apelată.
```

iii. Aranjarea restului parametrilor, după cum urmează. Stiva va contine aceste patru informatii, preluate din vîrf: XO+X si YO+Y pentru sfîrsitul ultimului arc; apoi XO si YO pentru începutul primului arc. Mem-0 va contine XO si mem-5 va contine YO. Mem-1 si mem-2 vor contine înlocuirile initiale pentru primul arc, U si V; iar mem-3 si mem-4 vor contine COS (G/A) si SIN (G/A) pentru utilizarea în bucla de desenare arc.

Formulele pentru U si V pot fi explicate în modul următor. In locul deplasării de-a lungul coardei finale, de lungime L, denumită, cu înlocuirile X si Y, se doreste deplasarea de-a lungul unei coarde initiale (care poate fi mai lungă) de lungime L*W, unde W=SIN(G/2*A) / SIN (G/2), cu deplasările X*W si Y*W, dar trecînd prin unghiul -(G/2 - G/2*A), deci cu adevăratele înlocuiri:

U = Y*W*SIN (G/2 - G/2*A) + X*W*COS (G/2 - G/2*A)
V = Y*W*COS (G/2 - G/2*A) - X*W*SIN (G/2 - G/2*A)

Aceste formule pot fi verificate printr-o diagramă, folosind întinderile normale ale lui COS (P-Q) si SIN (P-Q), unde Q=G/2 - G/2*A.

```
23C4            PUSH  BC                    Salvare contor-arc în B.
                RST   0028,FP-CALC          X,Y,SIN(G/2),Z
                DEFB  +02,delete            X,Y,SIN(G/2)
                DEFB  +E1,get-mem-1         X,Y,SIN(G/2)SIN(G/2*A)
                DEFB  +01,exchange          X,Y,SIN(G/2*A),SIN(G/2)
                DEFB  +05,division          X,Y,SIN(G/2*A)/SIN(G/2)=W
                DEFB  +C1,st-mem-1          (W este copiat în mem-1)
                DEFB  +02,delete            X,Y
                DEFB  +01,exchange          Y,X
                DEFB  +31,duplicate         Y,X,X
```

```
DEFB  +E1,get-mem-1       Y,X,X,W
DEFB  +04,multiply        Y,X,X*W
DEFB  +C2,st-mem-2        (X*W este copiat în mem-2)
DEFB  +02,delete          Y,X
DEFB  +01,exchange        X,Y
DEFB  +31,duplicate       X,Y,Y
DEFB  +E1,get-mem-1       X,Y,Y,W
DEFB  +04,multiply        X,Y,Y*W
DEFB  +E2,get-mem-2       X,Y,Y*W,X*W
DEFB  +E5,get-mem-5       X,Y,Y*W,X*W,G
DEFB  +E0,get-mem-0       X,Y,Y*W,X*W,G,G/A
DEFB  +03,subtract        X,Y,Y*W,X*W,G-G/A
DEFB  +A2,stk-half        X,Y,Y*W,X*W,G,G/A,1/2
DEFB  +04,multiply        X,Y,Y*W,X*W,G/2.G/2*A=F
DEFB  +31,duplicate       X,Y,Y*W,X*W,F,F
DEFB  1F,sin              X,Y,Y*W,X*W,F,SIN F
DEFB  +C5,st-mem-5        (SIN F este copiat în mem-5)
DEFB  +02,delete          X,Y,Y*W,X*W,F
DEFB  +20,cos             X,Y,Y*W,X*W,COS F
DEFB  +C0,st-mem-0        (COS F este copiat în mem-0)
DEFB  +02,delete          X,Y,Y*W,X*W
DEFB  +C2,st-mem-2        (X*W este copiat în mem-2)
DEFB  +02,delete          X,Y,Y*W
DEFB  +C1,st-mem-1        (Y*W este copiat în mem-1)
DEFB  +E5,get-mem-5       X,Y,Y*W,SIN F
DEFB  +04,multiply        X,Y,Y*W*SIN F
DEFB  +E0,get-mem-0       X,Y,Y*W*SIN F,X*W
DEFB  +E2,get-mem-2       X,Y,Y*W,SIN F,X*W,COS F
DEFB  +04,multiply        X,Y,Y*W*SIN F,X*W*COS F
DEFB  +0F,addition        X,Y,Y*W*SIN F+X*WCOS F = U
DEFB  +E1,get-mem-1       X,Y,U,Y*W
DEFB  +01,exchange        X,Y,Y*W,U
DEFB  +C1,st-mem-1        (U este copiat în mem-1)
DEFB  +02,delete          X,Y,Y*W
DEFB  +E0,get-mem-0       X,Y,Y*W,COS F
DEFB  +04,multiply        X,Y,Y*W*COS F
DEFB  +E2,get-mem-2       X,Y,Y*W*COS F,X*W
DEFB  +E5,get-mem-5       X,Y,Y*WCOS F,X*W,SIN F
DEFB  +04,multiply        X,Y,Y*WCOS F,X*W*SIN F
DEFB  +03,subtract        X,Y,Y*WCOS F-X*W*SIN F = V
DEFB  +C2,st-mem-2        (V este copiat în mem-2)
DEFB  +2A,abs             X,Y,V' (V' = ABS V)
DEFB  +E1,get-mem-1       X,Y,V',U
DEFB  +2A,abs             X,Y,V',U' (U' = ABS U)
DEFB  +0F,addition        X,Y,U'+V'
DEFB  +02,delete          X,Y
DEFB  +38,end-calc        (DE indică acum pe U'+V')
LD    A,(DE)              Se încarcă exponentul pentru
                          U'+V'
CP    +81                 Dacă U'+V' este mai mic decît
POP   BC                  1, doar se curăță stiva si se
JP    C,2477,LINE-DRAW    desenează linia de la X0,Y0
                          la X0+X, Y0+Y.
PUSH  BC                  Altfel se continuă cu
RST   0028,FP-CALC        parametrii: X,Y în stivă.
DEFB  +01,exchange        Y,X
DEFB  +38,end-calc
LD    A,(COORDS-lo)       Se aduce X0 în A si astfel în
CALL  2D28,STACK-A        stivă.
RST   0028,FP-CALC        Y,X,X0
DEFB  +C0,st-mem-0        (X0 este copiat în mem-0)
DEFB  +0F,addition        Y,X0+X
DEFB  +01,exchange        X0+X,Y
DEFB  +38,end-calc
LD    A,(COORDS-hi)       Se aduce Y0 în A si astfel în
CALL  2D28,STACK-A        stivă.
RST   0028,FP-CALC        X0+X,Y,Y0
DEFB  +C5,st-mem-5        (Y0 este copiat în mem-5)
DEFB  +0F,addition        X0+X,Y0+Y
DEFB  +E0,get-mem-0       X0+X,Y0+Y,X0
DEFB  +E5,get-mem-5       X0+X,Y0+Y,X0,Y0
DEFB  +38,end-calc
POP   BC                  Se reface contorul-arc în B.
```

iv. Bucla desenare arc. Este introdusă la 2439 cu coordonatele punctului de început în vîrful stivei, si cu înlocuirile initiale pentru primul arc în mem-1 si mem-2. Se foloseste trigonometria simplă pentru asigurarea faptului că toate arcele subsecventei vor fi desenate în punctele care le leagă în acelasi cerc ca primele două, subîntinzînd acelasi unghi la centru. Se poate arăta că

dacă 2 puncte X1,Y1 si X2,Y2 sînt legate pe un cerc si subîntind un unghi la centru N, care este si originea coordonatelor, atunci X2=X1*cos N.Y1*sin N, si Y2 = X1*SIN N + Y1*cos N. Dar din cauza faptului că originea este aici în coltul de jos din partea dreaptă a ecranului, bucla de desenare-arc aplică aici relatii de incrementare, denumind Un=Xn+1.Yn si Vn=Yn+1.Xn, care realizează rezultatul dorit. Stiva arată după (n+1) pasi prin buclă, că Xn si Yn sînt incrementati cu Un si Vn, după aceasta fiind obtinuti din Un-1 si Vn-1. Cele 4 valori din vîrful stivei la 2425 sînt, în DRAW, citind în sus, XO+X, YO+Y, Xn si Yn, dar pentru salvarea spatiului acestea nu sînt arătate pînă la 2439. Pentru valorile initiale în CIRCLE, vezi sfîrsitul lui CIRCLE, mai sus. De asemenea în CIRCLE unghiul G trebuie luat 2*PI.

```
2420 DRW-STEPS    DEC   B                 B numără trecerile prin buclă
                  JR    Z,245F,ARC-END    Salt cînd B ajunge la zero.
                  JR    2439,ARC-START    Salt la începutul buclei.
2425 ARC-LOOP     RST   0028,FP-CALC      (Vezi textul mai sus de
                                          stivă)
                  DEFB  +E1,get-mem-1     Un-1
                  DEFB  +31,duplicate     Un-1,Un-1
                  DEFB  +E3,get-mem-3     Un-1,Un-1,COS(G/A)
                  DEFB  +04,multiply      Un-1,Un-1*COS(G/A)
                  DEFB  +E2,get-mem-2     Un-1,Un-1*COS(G/A),Vn-1
                  DEFB  +E4,get-mem-4     Un-1,Un-1*COS(G/A),Vn-1,
                                          SIN(G/A)
                  DEFB  +04,multiply      Un-1,Un-1*COS(G/A),Vn-1*
                                          SIN(G/A)
                  DEFB  +03,subtract      Un-1,Un-1*COS(G/A)-Vn-1*
                                          SIN(G/A) = Un
                  DEFB  +C1,st-mem-1      (Un este copiat în mem-1)
                  DEFB  +02,delete        Un-1
                  DEFB  +E4,get-mem-4     Un-1,SIN(G/A)
                  DEFB  +04,multiply      Un-1*SIN(G/A)
                  DEFB  +E2,get-mem-2     Un-1*COS(G/A),Vn-1
                  DEFB  +E3,get-mem-3     Un-1*SIN(G/A),Vn-1,COS(G/A)
                  DEFB  +04,multiply      Un-1*SIN(G/A),Vn-1*COS(G/A)
                  DEFB  +0F,addition      Un-1*SIN(G/A)+Vn-1*COS(G/A)=
                                          Vn
                  DEFB  +C2,st-mem-2      (Vn este copiat în mem-2)
                  DEFB  +02,delete        (Cum s-a notat si în text,
                  DEFB  +38,end-calc      stiva contine de fapt XO+X,
                                          YO+Y, Xn si Yn)
2439 ARC-START    PUSH  BC                Salvare contor-arc.
                  RST   0028,FP-CALC      XO+X,YO+Y
                  DEFB  +C0,st-mem-0      (Yn este copiat în mem-0)
                  DEFB  +02,delete        XO+X,YO+Y,Xn
                  DEFB  +E1,get-mem-1     XO+X,YO+Y,Xn,Un
                  DEFB  +0F,addition      XO+X,YO+Y,Xn+Un,Xn+1
                  DEFB  +31,duplicate     XO+X,YO+Y,Xn+1,Xn+1
                  DEFB  +38,end-calc      Urmează Xn', valoarea
                                          aproximată a lui Xn obtinută
                                          prin subrutina de desenare-
                                          linie, care este copiată în
                                          A
                  LD    A,(COORDS-lo)     si de aici în stivă.
                  CALL  2D28,STACK-A      XO+X,YO+Y,Xn+1,Xn'
                  RST   0028,FP-CALC      XO+X,YO+Y,Xn+1,Xn+1,Xn'-Xn'=
                  DEFB  +03,subtract      Un'
                  DEFB  +E0,get-mem-0     XO+X,YO+Y,X+1,Un',Yn
                  DEFB  +E2,get-mem-2     XO+X,YO+Y,Xn+1,Un',Yn,Vn
                  DEFB  +0F,addition      XO+X,YO+Y,Xn+1,Un',Yn+Vn=Yn+1
                  DEFB  +C0,st-mem-0      (Yn+1 este copiat în mem-0)
                  DEFB  +01,exchange      XO+X,YO+Y,Xn+1,Yn+1,Un'
                  DEFB  +E0,get-mem-0     XO+X,YO+Y,Xn+1,Yn+1,Un',Yn+1
                  DEFB  +38,end-calc
                  LD    A,(COORDS-hi)     Yn', aproximat ca si Xn',este
                  CALL  2D28,STACK-A      copiat în A si de aici în
                                          stivă.
                  RST   0028,FP-CALC      XO+X,YO+Y,Xn+1,Yn+1,Un',Yn+1,
                                          Yn'
                  DEFB  +03,subtract      XO+X,YO+Y,Xn+1,Yn+1,Un',Vn'
                  DEFB  +38,end-calc
                  CALL  24B7,DRAW-LINE    Este desenat următorul 'arc'.
                  POP   BC                Restituire contor-arc.
                  DJNZ  2425,ARC-LOOP     Salt dacă trebuie desenate
                                          mai multe arcuri.
245F ARC-END      RST   0028,FP-CALC      Coordonatele sfîrsitului
                  DEFB  +02,delete        ultimului arc care a fost
                  DEFB  +02,delete        desenat sînt acum sterse din
                                          stivă.
```

```
                DEFB  +01,exchange        YO+Y,XO+X
                DEFB  +38,end-calc
                LD    A,(COORDS-lo)       Coordonata X a sfîrsitului
                CALL  2D28,STACK-A        ultimului arc care a fost
                RST   0028,FP-CALC        desenat, numit Xz', se
                                          copiază în stivă.
                DEFB  +03,subtract        YO+Y,XO+X-Xz'
                DEFB  +01,exchange        XO+X-Xz',YO+Y
                DEFB  +38,end-calc
                LD    A,(COORDS-hi)       S-a obtinut coordonata Y.
                CALL  2D28,STACK-A
                RST   0028,FP-CALC        XO+X-Xz',YO+Y,Yz'
                DEFB  +03,subtract        XO+X-Xz',YO+Y-Yz'
                DEFB  +38,end-calc
2477 LINE-DRAW  CALL  2487,DRAW-LINE      Arcul final este desenat pînă
                                          ajunge la XO+X,YO+Y (sau să
                                          închidă cercul).
                JP    0D4D,TEMPS          Iesire, setînd culorile
                                          temporare.
```

THE 'INITIAL PARAMETERS' SUBROUTINE (SUBRUTINA 'PARAMETRII INITIALI')
Această subrutină este apelată si de CIRCLE si de DRAW pentru a seta parametrii lor initiali. Ea este apelată de CIRCLE cu X, Y si raza Z în vîrful stivei, citind partea de sus. Ea este apelată de DRAW cu ai săi X, Y, SIN (G/2) si Z, cum au fost definiti în DRAW i, deasupra, în vîrful stivei. In cele ce urmează stiva este arătată doar de la Z în sus.

Subrutina readuce în B contorul arc, A cum a fost explicat si la începutul lui DRAW si la începutul lui CIRCLE si în mem-0 la mem-5 cantitătile G/A, SIN (G/2*A),0,COS (G/A), SIN (G/A) si G. Pentru un cerc, G trebuie luat egal cu 2*PI.

```
247D CD-PRMS1   RST   0028,FP-CALC       Z
                DEFB  +31,duplicate       Z,Z
                DEFB  +28,sqr             Z,SQR Z
                DEFB  +34,stk-data        Z,SQR Z,2
                DEFB  +32,exponent+82
                DEFB  +00,(+00,+00,+00)
                DEFB  +01,exchange        Z,2,SQR Z
                DEFB  +05,division        Z,2/SQR Z
                DEFB  +E5,get-mem-5       Z,2/SQR Z,G
                DEFB  +01,exchange        Z,G,2/SQR Z
                DEFB  +05,division        Z,G*SQR Z/2
                DEFB  +2A,abs             Z,G'*SQR Z/2 (G' = MOD G)
                DEFB  +38,end-calc        Z,G'*SQR Z/2 = A1
                CALL  2DD5,FP-TO-A        A1 la A din stivă, dacă este
                                          posibil.
                JR    C,2495,USE-252      Dacă A1 ciclează de 256 ori
                                          sau mai mult, se utilizează
                                          256.
                AND   +FC                 4*INT (A1/4) la A.
                ADD   A,+04               Se adună 4, dînd contorul-arc
                                          A.
                JR    NC,2497,DRAW-SAVE   Salt dacă-încă este sub 256.
2495 USE-252    LD    A,+FC               Aici, doar utilizarea 256
                                          zecimal.
                PUSH  AF                  Acum salvare contor-arc.
                CALL  2D28,STACK-A        Copierea lui în stiva
                                          calculatorului.
                RST   0028,FP-CALC        Z,A
                DEFB  +E5,get-mem-5       Z,A,G
                DEFB  +01,exchange        Z,G,A
                DEFB  +05,division        Z,G/A
                DEFB  +31,duplicate       Z,G/A,G/A
                DEFB  +1F,sin             Z,G/A,SIN (G/A)
                DEFB  +C4,st-mem-4        (SIN (G/A) se copiază în
                                          mem-4)
                DEFB  +02,delete          Z,G/A
                DEFB  +31,duplicate       Z,G/A,G/A
                DEFB  +A2,stk-half        Z,G/A,G/A,0.5
                DEFB  +04,multiply        Z,G/A,G/2*A
                DEFB  +1F,sin             Z,G/A,SIN (G/2*A)
                DEFB  +C1,st-mem-1        (SIN (G/2*A) se copiază în
                                          mem-1)
                DEFB  +01,exchange        Z,SIN (G/2*A),G/A
                DEFB  +C0,st-mem-0        (G/A se copiază în mem-0)
                DEFB  +02,delete          Z,SIN (G/2*A) = S
                DEFB  +31,duplicate       Z,S,S
                DEFB  +04,multiply        Z,S*S
                DEFB  +31,duplicate       Z,S*S,S*S
```

```
                DEFB  +0F,addition        Z,2*S*S
                DEFB  +A1,stk-one         Z,2*S*S,1
                DEFB  +03,subtract        Z,2*S*S-1
                DEFB  +1B,negate          Z,1-2*S*S = COS (G/A)
                DEFB  +C3,st-mem-3        (COS (G/A) se copiază în
                                          mem-4)
                DEFB  +02,delete          Z
                DEFB  +38,end-calc
                POP   BC                  Readucerea contorului-arc în
                                          B.
                RET                       Terminare.
```

THE LINE-DRAWING SUBROUTINE (SUBRUTINA TRASARE LINIE)
Această subrutină este apelată de DRAW pentru a desena o aproximare la o linie dreaptă din punctul X0,Y0 continut în COORDS la punctul X0+X,Y0+Y, unde incrementele X si Y sînt în vîrful stivei calculatorului. Subrutina a fost proiectată initial pentru ZX80 si ZX81 8K ROM, si este descrisă de un program BASIC la pagina 121 din manualul ZX81. De asemenea este ilustrată si aici în programul Circle din anexă.

Metoda este de a interpătrunde atîti pasi orizontal sau vertical cîti sînt necesari de-a lungul setului de bază al pasilor diagonali, folosind un algoritm care aseazâ pasii orizontal sau vartical, după cum este posibil.

```
24B7 DRAW-LINE    CALL  2307,STK-TO-BC     ABS Y în B; ABS X în C;
                                          SGN Y în D; SGN X în C.
                  LD    A,C               salt dacă ABS X este mai mare
                  CP    B                 sau egal cu ABS Y, astfel
                  JR    NC,24C4,DL-X-GE-Y încît cel mai mic merge în L,
                  LD    L,C               si cel mai mare merge în H.
                  PUSH  DE                Salvare pas diagonal (±1,±1)
                                          în DE.
                  XOR   A                 Inserare pas vertical (±1,0)
                  LD    E,A               în DE (D contine SGN Y).
                  JR    24CB,DL-LARGER    Acum salt pentru a seta H.
24C4 DL-X-GE-Y    OR    C                 Revenire dacă si ABS X si
                  RET   Z                 ABS Y sînt ambii zero.
                  LD    L,B               Cel mai mic (aici ABS Y)
                                          trece în L.
                  LD    B,C               In acest caz, ABS X trece în
                                          B din H.
                  PUSH  DE                Salvare pas diagonal.
                  LD    D,+00             Pas orizonatal (0,±1) în DE.
24CB DL-LARGER    LD    H,B               Cel mai mare dintre ABS X si
                                          ABS Y trece acum în H.
```

Algoritmul începe aici. Cel mai mare dintre ABS X si ABS Y, numit H, este pus în A si redus la INT (H/2). Pasii H-L orizontali sau verticali si pasii diagonali L sînt făcuti (unde L este mai mic decît ABS X si ABS Y) în modul următor: L este adunat la A; Dacă acum A este mai mare sau egal cu H, el este redus cu H si se face un pas diagonal; altfel se face un pas vertical sau orizontal. Aceasta se repetă de H ori (B contine pe H). De notat faptul că în acelasi timp schimbarea registrilor H' si L' este utilizată pentru a păstra COORDS.

```
                  LD    A,B               B în A la fel de bine ca si
                                          în H.
                  RRA                     A începe la INT (H/2).
24CE D-L-LOOP     ADD   A,L               L este adunat la A.
                  JR    C,24D4,D-L-DIAG   Dacă este 256 sau mai mult,
                                          salt la pas diagonal.
                  CP    H                 Dacă A este mai mic decît H,
                  JR    C,24DB,D-L-HR-VT  salt pentru pas orizontal sau
                                          vertical.
24D4 D-L-DIAG     SUB   H                 Se reduce A cu H.
                  LD    C,A               Se reface în C.
                  EXX                     Acum se utilizează schimbarea
                                          registrilor.
                  POP   BC                Pas diagonal la B'C'.
                  PUSH  BC                Si acesta este salvat.
                  JR    24DF,D-L-STEP     Salt pentru a executa pasul.
24DB D-L-HR-VT    LD    C,A               Se salvează A (neredus) în C.
                  PUSH  DE                Se trece în stivă pentru
                                          scurt timp.
                  EXX                     Se interschimbă registrii.
                  POP   BC                Se trece în B'C' acum.
24DF D-L-STEEP    LD    HL,(COORDS)       Acum se face pasul: mai întîi
                                          COORDS în H'L' ca pun
început.
                  LD    A,B               Pas Y de la B' la A.
```

```
                ADD     A,H             Se adună în H'.
                LD      B,A             Rezultatul în B.
                LD      A,C             Acum pas X; îi va fi testat
                INC     A               rangul (Y trebuie testat în
                                        PLOT).
                ADD     A,L             Se adună L' la C' în A. Salt
                JR      C,24F7,D-L-RANGE la trasport pentru a testa
                                        mai departe.
                JR      Z,24F9,REPORT-B Zero după 'nu este transport'
                                        denotă poziția X-1, afară
                                        din rang.
24EC D-L-PLOT   DEC     A               Se reface valoarea adevărată
                                        a lui A.
                LD      C,A             Valoarea pentru reprezentarea
                                        grafică în C'.
                CALL    22E5,PLOT-SUB   Reprezentare pas.
                EXX                     Se refac principalii registri
                LD      A,C             C înapoi în A pentru a
                                        continua algoritmul.
                DJNZ    24CE,D-L-LOOP   Salt înapoi pentru B pasi (H
                                        pasi).
                POP     DE              Se sterge stiva
                                        calculatorului.
                RET                     Terminare.
24F7 D-L-RANGE  JR      24EC,D-L-PLOT   Zero după transport denotă
                                        poziția X 255, în rang.

Prezentarea B - Intreg afară din domeniu.

24F9 REPORT-B   RST     0008,ERROR-1    Se apelează rutina de tratare
                DEFB    +0A             eroare.
```

EXPRESSION EVALUTION

THE 'SCANNING' SUBROUTINE

Această subrutină este utilizată pentru a realiza evaluarea rezultatului 'următoarei expresii'.

Rezultatul este returnat ca 'ultima valoare' în stiva calculatorului. Pentru un rezultat numeric, ultima valoare va fi numărul actual în virgulă mobilă. După cum, pentru un rezultat sir ultima valoare va consta dintr-un set de parametrii. Primul din cinci octeti este nespecificat, al doilea si al treilea octet contin adresa de început a sirului si al patrulea si al cincilea octet contin lungimea sirului.

Bitul 6 al FLAGS este setat în cazul unui rezultat numeric si resetat pentru un rezultat sir.

Cînd o următoare expresie constă dintr-un singur operand, de exemplu simplu valoarea obtinută prin evaluarea operandului.

După cum, cînd următoarea expresie contine o functie si un operand, de exemplu ...CHR$ A...,...NOT A...,SIN 1..., codul operatiei functiei este stocat în stiva masinii pînă cînd se calculează ultima valoare a operandului. Această ultimă valoare este apoi supusă unei operatii corespunzătoare pentru a da o nouă ultimă valoare.

In cazul în care trebuie efectuată o operatie logică sau aritmetică, de exemplu ...A+B...,A*B...,...A=B..., atunci ambele valori ultime ale primului argument si ale codului operatiei trebuie păstrate pînă se găseste ultima valoare a celui de al doilea argument. Intr-adevăr, calculul ultimei valori a celui de al doilea argument poate de asemenea să includă stocarea ultimelor valori si a codurilor operatiei pînă cînd s-a terminat calculul.

De aceea se poate arăta că atunci cînd se evaluează o expresie complexă, ce de exemplu ...CHR$ (+A-26*INT ((T+A)/26)+65)..., trebuie realizată o ierarhie a operatiilor care mai sînt de prelucrat pînă cînd se ajunge în punctul din care se poate renunta la ierarhie pentru a găsi în final ultima valoare.

Fiecare cod de operatiei are asociat un cod corespunzător al prioritătii si operatiile de prioritate superioară se execută întotdeauna înaintea celor de prioritate inferioară.

Subrutina începe cu registrul A setat astfel încît să conțină primul caracter al expresiei si marcatorul prioritătii de început - zero - încărcat în stiva masinii.

```
24FB SCANNING     RST   0018,GET-CHAR        Se aduce primul caracter.
                  LD    B,+00                Marcatorul prioritătii de
                  PUSH  BC                   început este stocat.
24FF S-LOOP-1     LD    C,A                  Punctul principal de intrare.
                  LD    HL,+2596             Index în tabelul de baleiere
                  CALL  160C,INDEXER         functii cu codul în C.
                  LD    A,C                  Se reface codul în A.
                  JP    NC,2684,S-ALPHNUM    Salt dacă codul nu s-a găsit
                                             în tabel.
                  LD    B,+00                Se foloseste intrarea găsită
                  LD    C,(HL)               în tabel pentru a construi
                  ADD   HL,BC                adresa cerută în HL si salt
                  JP    (HL)                 la aceasta.
```

Urmează patru subrutine; ele sînt apelate de rutine din tabelul de baleiere functie. Prima, 'subrutina de baleiere a limitelor', este folosită de S-QUOTE pentru a verifica dacă limita fiecărui sir coincide cu cealaltă.

```
250F S-QUOTE-S    CALL  0074,CH-ADD+1        Se indică următorul caracter.
                  INC   BC                   Se incrementează cu 1
                                             contorul lungimii.
                  CP    +0D                  Este un 'carriage return'?
                  JP    Z,1C8A,REPORT-C      Prezentare eroare dacă este
                                             asa.
                  CP    +22                  Este un alt '"'?
                  JR    NZ,250F,S-QUITE-S    Salt înapoi dacă nu este.
                  CALL  0074,CH-ADD+1        Se indică următorul caracter;
                  CP    +22                  se setează fanionul zero dacă
                                             este un alt '"'.
                  RET                        Sfîrsit.
```

Următoarea subrutină, subrutina 'baleiere: două coordonate', este apelată de S-SCREEN$, S-ATTR si S-POINT pentru a se asigura că cele două coordonate cerute sînt date în forma lor specifică.

```
2522 S-2-COORD    RST   0020,NEXT-CHAR       Se aduce următorul caracter.
                  CP    +28                  Este '('?
                  JR    NZ,252D,S-RPORT-C    Prezentare eroare dacă nu
                                             este.
                  CALL  1C79,NEXT-2NUM       Coordonatele în stiva
                                             calculatorului.
                  RST   0018,GET-CHAR        Se aduce caracterul curent.
```

```
                CP    +29                       Este ')'?
252D S-RPORT-C  JP    NZ,1C8A,REPORT-C          Prezentare eroare dacă nu
                                                este.
```

THE 'SYNTAX-Z' SUBROUTINE (SUBRUTINA 'SYNTAX-Z')
In acest punct se interpolează subrutina 'SYNTAX-Z'.Este apelată de 32 de ori, cu salvarea doar a unui octet la fiecare apelare. Un simplu test al bitului 7 al FLAGS va da resetarea fanionul zero de-a lungul executiei si setarea în timpul verificării sintaxei.

```
2530 SYNTAX-Z   BIT   7,(FLAGS)                 Testare bit 7 al FLAGS.
                RET                             Sfîrsit.
```

Următoarea subrutină este 'subrutina baleiere SCREEN$', care este folosită de S-SCREEN$ pentru a găsi caracterul ce apare pe ecran la linia x, coloana y. Aceasta doar caută caracterul setat 'indicat' în CHARS.

Notă: Acesta este în mod normal unul din caracterele +20 (spatiu) la +7F (©), altfel utilizatorul poate modifica CHARS egalînd cu alte caractere, incluzînd caracterele grafice definite de utilizator.

```
2535 S-SCRN$-S  CALL  2307,STK-TO-BC            x în C; y în B; 0 <= x <= 23
                LD    HL,(CHARS)                zecimal; 0 <= y <= 31 zecimal
                LD    DE,+0100                  CHARS plus 256 zecimal face
                ADD   HL,DE                     ca HL să indice caracterul
                                                setat.
                LD    A,C                       x se copiază în A.
                RRCA                            Numărul 32 (zecimal) * (x mod
                RRCA                            8) + y este format în A si
                                                copiat în E.
                RRCA                            Acesta este octetul inferior
                AND   +E0                       al adresei ecran cerute.
                XOR   B
                LD    E,A
                LD    A,C                       Se copiază din nou x în A.
                AND   +18                       Acum numărul 64 (zecimal) +
                XOR   +40                       8*INT (x/8) este introdus în
                                                D.
                LD    D,A                       DE contine acum adresa
                                                ecranului.
                LD    B,+60                     B numără cele 96 de caractere
25AF S-SCRN-LP  PUSH  BC                        Salvare contor.
                PUSH  DE                        Salvare indicator ecran.
                PUSH  HL                        Salvare indicator caracter
                                                setat.
                LD    A,(DE)                    Se aduce primul rînd al
                                                caracterului ecran.
                XOR   (HL)                      Se compară cu rîndul
                                                caracterului setat.
                JR    Z,255A,S-SC-MATCH         Salt dacă s-a găsit egalitate
                                                directă.
                INC   A                         Acum se face test egalitate
                                                cu caracterul invers (se pune
                                                +00 în A în loc de +FF).
                JR    NZ,2573,S-SCR-NXT         Salt dacă nu s-a găsit nici o
                                                egalitate.
                DEC   A                         Se reface +FF în A.
255A S-SC-MATCH LD    C,A                       Inversare stare (+00 sau +FF)
                                                în C.
                LD    B,+07                     B numără celelalte 7 rînduri.
255D S-SC-ROWS  INC   D                         Se mută DE în celălalt rînd
                                                (se adună 256 zecimal).
                INC   HL                        Se mută HL în rîndul următor
                                                (aceasta înseamnă următorul
                                                octet).
                LD    A,(DE)                    Se aduce rîndul de ecran.
                XOR   (HL)                      Se compară cu rîndul din ROM.
                XOR   C                         Se include starea inversă.
                JR    NZ,2573,S-SCR-NXT         Salt dacă rîndul reuseste
                                                egalarea.
                DJNZ  255D,S-SC-ROWS            Salt pînă s-au dat toate
                                                rîndurile.
                POP   BC                        Se înlătură indicatorul
                                                caracterului setat.
                POP   BC                        Si indicatorul ecran.
                POP   BC                        Contor final în BC.
                LD    A,+80                     Ultimul cod caracter este
                                                setat plus unu.
                SUB   B                         Acum A contine codul cerut.
```

```
                LD    BC,+0001          Acum este nevoie de un spatiu
                                        în spatiul de lucru.
                RST   0030,BC-SPACES    Se face spatiul.
                LD    (DE),A            Se pune caracterul în el.
                JR    257D,S-SCR-STO    Salt pentru stocarea
                                        caracterului.
2573 S-SCR-NXT  POP   HL                Restituire indicator caracter
                                        setat.
                LD    DE,+0008          Se trece cu 8 octeti, la
                ADD   HL,DE             următorul caracter din set.
                POP   DE                Restituire indicator ecran.
                POP   BC                Si contor.
                DJNZ  254F,S-SCRN-LP    Ciclare înapoi pentru cele 96
                                        de caractere.
                LD    C,B               Se încarcă sir gol (lungime
                                        zero).
257D S-SCR-STO  JP    2AB2,STK-STO-$    Salt pentru stocarea
                                        caracterului egal, sau a
                                        sirului nul dacă nu s-a găsit
                                        nici o egalitate.
```

Notă: Această iesire, prin STK-STO-$, este o greșală si conduce la 'stocarea dublă' a sirului rezultat (vezi S-STRING, 25DB). Linia de instructie trebuie să fie 'RET'.

Ultima din aceste patru subrutine este 'subrutina baleiere atribute'. Ea este apelată de S-ATTR pentru a returna valoarea lui ATTR (x,y) care codifică atributele liniei x,coloanei y pe ecranul televizorului.

```
2580 S-ATTR-S   CALL  2307,STK-TO-BC    x în C, y în B. Iarăsi, 0 <=
                LD    A,C               x <= 23 zecimal, 0 <= y <= 31
                                        zecimal.
                RRCA                    Se copiază x în A si numărul
                RRCA                    32 (zecimal)*x(mod 8)+y este
                RRCA                    format în A si copiat în E.
                LD    C,A               32*x(mod 8)+INT (x/8) este de
                AND   +E0               asemenea copiat în C.
                XOR   B
                LD    L,A               L contine octetul inferior al
                                        adresei atributului.
                LD    A,C               Se copiază în A 32*x(mod 8)+
                                        INT (x/8).
                AND   +03               88 (zecimal)+INT (x/8) este
                XOR   +58               format în A si copiat în H.
                LD    H,A               H contine octetul superior al
                                        adresei atributului.
                LD    A,(HL)            Octetul atribut este copiat
                                        în A.
                JP    2D28,STACK-A      Iesire, stocînd octetul cerut
```

THE SCANNING FUNCTION TABLE (TABELUL FUNCTIEI DE BALEIERE)
Acest tabel contine 8 functii si 4 operatori. Acesta încorporează 5 noi functii Spectrum si prevede o cale curată de accesare a unor functii si operatori care deja există în ZX81.

locatia	cod	deplasament	nume	adresa rutinei tratate
2596	22	1C	S-QUOTE	25B3
2598	28	4F	S-BRACKET	25E8
259A	2E	F2	S-DECIMAL	268D
259C	2B	12	S-U-PLUS	25AF
259E	A8	56	S-FN	25F5
25A0	A5	57	S-RND	25F8
25A2	A7	84	S-PI	2627
25A4	A6	8F	S-INKEY$	2634
25A6	C4	E6	S-BIN (EQU,S-DECIMAL)	2680
25A9	AA	BF	S-SCREEN$	2668
25AA	AB	C7	S-ATTR	2672
25AC	A9	CE	S-POINT	2678
25AE	00			indicator de sfîrsit

THE SCANNING FUNCTION ROUTINES (RUTINELE FUNCTIEI BALEIERE)

```
25AF S-U-PLUS   RST   0020,NEXT-CHAR    Pentru un singur plus, pur si
                JP    24FF,S-LOOP-1     simplu se trece la următorul
                                        caracter si salt înapoi la
```

punctul principal de intrare al lui SCANNING.

'Rutina baleiere LIMITE': Această rutină lucrează cu limite de sir, sau simple ca "name" sau mai complexe ca "a""white""lie"" sau aparent redundante VAL$ """a""".

```
25B3 S-QUOTE        RST   0018,GET-CHAR      Se aduce caracterul curent.
                    INC   HL                 Se indică începutul sirului.
                    PUSH  HL                 Se salvează adresa de început
                    LD    BC,+0000           Se setează lungimea cu zero.
                    CALL  250F,S-QUOTE-S     Se apelează rutina de
                                             'coincidentă'.
                    JR    NZ,25D9,S-Q-PRMS   Salt dacă zero este resetat -
                                             nu mai sînt limite.
25BE S-Q-AGAIN      CALL  250F,S-QUOTE-S     Se apelează din nou pentru a
                                             treia limită.
                    JR    Z,25BE,S-Q-AGAIN   Si din nou pentru a cincea, a
                                             saptea, etc.
                    CALL  2530,SYNTAX-Z      Dacă se testează sintaxa, se
                    JR    Z,25D9,S-Q-PRMS    execută un salt pentru a
                                             reseta bitul 6 al lui FLAGS
                                             si a continua baleierea.
                    RST   0030,BC-SPACES     Se creează spatiu în spatiul
                                             de lucru pentru sir si pentru
                                             limita terminală.
                    POP   HL                 Se aduce indicatorul la
                                             început.
                    PUSH  DE                 Se salvează indicatorul în
                                             primul spatiu.
25CB S-Q-COPY       LD    A,(HL)             Se aduce un caracter din sir.
                    INC   HL                 Se indică următorul.
                    LD    (DE),A             Se copiază ultimul în spatiul
                                             de lucru.
                    INC   DE                 Se indică următorul spatiu.
                    CP    +22                Este ultimul caracter un '"'?
                    JR    NZ,25CB,S-Q-COPY   Dacă nu este, se execută salt
                                             pentru a copia următorul
                                             caracter.
                    LD    A,(HL)             Dar dacă a fost, nu se
                    INC   HL                 copiază următorul; dacă
                    CP    +22                următorul este un '"', se
                    JR    Z,25CB,S-Q-COPY    sare pentru a-l copia pe
                                             următorul de după el; altfel
                                             se termină cu copierea.
25D9,S-Q-PRMS       DEC   BC                 Se aduce adevărata lungime în
                                             BC.
```

De notat că prima limită nu a fost contorizată în lungime; limita finală a fost, si este acum înlăturată. In interiorul sirului, prima, a treia, a cincea, etc., limită au fost contorizate dar a doua, a patra, etc., limită nu au fost contorizate.

```
                    POP   DE                 Se readuce începutul sirului
                                             copiat.
25D8 S-STRING       LD    HL,+5C3B           FLAGS; acest punct de intrare
                    RES   6,(HL)             este folosit întotdeauna când
                    BIT   7,(HL)             bitul 6 trebuie resetat si
                    CALL  NZ,2AB2,STK-STO-$  un sir stocat dacă se execută
                                             o linie. Aceasta se
                                             realizează acum.
                    JP    2712,S-CONT-2      Salt pentru a continua
                                             baleierea liniei.
```

De notat că la copierea sirului în spatiul de lucru, fiecare două perechi de limite ale sirului din interiorul sirului (" ") vor fi reduse la o pereche de limite ale sirului (").

```
25E8 S-BRACKET      RST   0020,NEXT-CHAR     'Rutina de baleiere BRACKET'
                    CALL  24FB,SCANNING      aduce doar următorul caracter
                    CP    +29                si apelează recursiv
                                             SCANNING.
                    JP    NZ,1C8A,REPORT-C   Se prezintă eroare dacă nu
                    RST   0020,NEXT-CHAR     este parantează; apoi se
                                             continuă baleierea.
                    JP    2712,S-CONT-2
25F5 S-FN           JP    27BD,S-FN-SBRN     'Rutina de baleiere FN'.
```

Această rutină, pentru functiile definite de utilizator, execută doar salt la

```
'subrutina de baleiere FN'.

25F8 S-RND      CALL  2530,SYNTAX-Z       Numai dacă s-a verificat
                JR    Z,2625,S-RND-END    sintaxa, se sare pentru a
                                          calcula un număr aleator.
                LD    BC,(SEED)           Se aduce valoarea curentă a
                                          lui SEED.
                CALL  2D28,STACK-BC       Aceasta se pune în stiva
                                          calculatorului.
                RST   0028,FP-CALC        Acum se foloseste
                                          calculatorul.
                DEFB  +A1,stk-one         'Ultima valoare' este acum
                DEFB  +0F,addition        SEED+1.
                DEFB  +34,stk-data        Se pune numărul zecimal 75
                DEFB  +37,exponent+87     în stiva calculatorului.
                DEFB  +16,(+00,+00,+00)
                DEFB  +04,multiply        'Ultima valoare' (SEED+1)*75.
                DEFB  +34,stk-data        Se observă STACK LITERALS
                DEFB  +80,(four bytes)    pentru a vedea cîti octeti
                DEFB  +41,exponent+91     se ocupă pentru a pune
                DEFB  +00,+00,+80,(+00)   numărul zecimal 65537 în
                                          stiva calculatorului.
                DEFB  +32,n-mod-m         Se împarte (SEED+1)*75 cu
                                          65537 pentru a obtine un
                                          'rest' si un 'răpuns'.
                DEFB  +02,delete          Se înlătură 'răspunsul'.
                DEFB  +A1,stk-one         'Ultima valoare' este acum
                DEFB  +03,subtract        'rest' - 1.
                DEFB  +31,duplicate       Se face o copie a 'ultimei
                                          valori'.
                DEFB  +38,end-calc        Calculul s-a terminat.
                CALL  2DA2,FP-TO-BC       Se foloseste 'ultima valoare'
                LD    (SEED),BC           pentru a da noua valoare lui
                                          SEED.
                LD    A,(HL)              Se aduce exponentul 'ultimei
                                          valori'.
                AND   A                   Salt înainte dacă exponentul
                JR    Z,2625,S-RND-END    este zero.
                SUB   +10                 Se reduce exponentul, aceasta
                LD    (HL),A              însemnînd împărtirea 'ultimei
                                          valori' cu 65536 pentru a
                                          obtine 'ultima valoare'
                                          cerută.
2625 S-RND-END  JR    2630,S-PI-END       Salt peste rutina 'PI'.

'Rutina de baleiere PI': numai după ce s-a verificat sintaxa, valoarea lui
'PI' este calculată si se formează 'ultima valoare' în stiva calculatorului.

2627 S-PI       CALL  2530,SYNTAX-Z       Test de verificare a sintaxei
                JR    Z,2630,S-PI-END     Salt dacă se cere.
                RST   0028,FP-CALC        Acum se utilizează
                                          calculatorul.
                DEFB  +A3,stk-pi/2        Valoarea lui PI/2 este pusă
                DEFB  +38,end-calc        în stiva calculatorului ca
                                          'ultima valoare'.
                INC   (HL)                Exponentul este incrementat,
                                          astfel dublînd 'ultima
                                          valoare' care dă PI.
2630 S-PI-END   RST   0020,NEXT-CHAR      Se trece la următorul
                                          caracter.
                JP    26C3,S-NUMERIC      Salt înainte.
2634 S-INKEY-$  LD    BC,+105A            Prioritate +10 hex, codul
                RST   0020,NEXT-CHAR      operatiei +5A pentru
                                          subrutina 'read-in'.
                CP    +23                 Salt dacă următorul caracter
                                          este '#'.
                JP    Z,270D,S-PUSH-PO    Va fi un argument numeric.
                LD    HL,+5C3B            Este FLAGS.
                RES   6,(HL)              Se resetează bitul 6 în cazul
                                          unui rezultat sir.
                BIT   7,(HL)              Test pentru verificarea
                                          sintaxei.
                JR    Z,2665,S-INKEY-$    Salt dacă se cere.
                CALL  028E,KEY-SCAN       Se aduce valoarea unei taste
                                          în DE.
                LD    C,+00               Pregătire sir gol; acesta se
                JR    NZ,2660,S-IK$-STK   stochează dacă sînt apăsate
                                          prea multe taste.
                CALL  031E,K-TEST         Se testează valoarea tastei;
```

```
                JR    NC,2660,S-IK$-STK     se stochează sir gol dacă nu
                                            este satisfăcătoare.
                DEC   D                     Se pune +FF în D pentru mod L
                                            (bitul 3 setat).
                LD    E,A                   Valoarea tastei se pune în E
                                            pentru a fi decodificată.
                CALL  0030,K-DECODE         Se decodifică valoarea tastei
                PUSH  AF                    Se salvează pentru scurt timp
                                            valoarea ASCII.
                LD    BC,+0001              Este necesar un spatiu în
                                            spatiul de lucru.
                RST   0030,BC-SPACES        Acesta se face acum.
                POP   AF                    Se readuce valoarea ASCII.
                LD    (DE),A                Pregătire pentru stocare dacă
                                            este un sir.
                LD    C,+01                 Lungimea sa este unu.
2660 S-IK$-STK  LD    B,+00                 Se completează parametrul
                                            lungime.
                CALL  2AB2,STK-STO-$        Se stochează sirul cerut.
2665 S-INK$-EN  JP    2712,S-CONT-2         Salt înainte.
2668 S-SCREEN$  CALL  2522,S-2-COORD         Se verifică dacă s-au dat
                                            cele 2 coordonate.
                CALL  NZ,2535,S-SCRN$-S     Se apelează subrutina numai
                RST   0020,NEXT-CHAR        dacă s-a verificat sintaxa;
                JP    25DB,S-STRING         apoi se aduce următorul
                                            caracter si salt înapoi.
2672 S-ATTR     CALL  2522,S-2-COORD         Se verifică dacă s-au dat
                                            cele 2 coordonate.
                CALL  NZ,2580,S-ATTR-S      Se apelează subrutina numai
                RST   0020,NEXT-CHAR        dacă s-a verificat sintaxa;
                JR    26C3,S-NUMERIC        apoi se aduce parametrul
                                            următor si salt înapoi.
267B S-POINT    CALL  2522,S-2-COORD         Se verifică dacă s-au dat
                                            cele 2 coordonate.
                CALL  NZ,22CB,POINT-SUB     Se apelează subrutina numai
                RST   0020,NEXT-CHAR        dacă s-a verificat sintaxa;
                JR    26C3,S-NUMERIC        apoi se aduce caracterul
                                            următor si salt înainte.
2684 S-ALPAHNUM CALL  2C88,ALPHANUM         Este caracter alfanumeric?
                JR    NC,26DF,S-NEGATE      Salt dacă nu este literă sau
                                            cifră.
                CP    +41                   Acum salt dacă este literă;
                JR    NC,26C9,S-LETTER      altfel se continuă în
                                            S-DECIMAL.
```

'Rutina baleiere DECIMAL' care urmează lucrează cu punct zecimal sau cu un număr care începe cu o cifră. De asemenea are grijă de expresia 'BIN', cu care lucrează în subrutina zecimal cu virgulă mobilă.

```
268D S-DECIMAL  CALL  2530,SYNTAX-Z         Salt înainte dacă s-a
     (EQU,S-BIN) JR   NZ,26B5,S-STK-DEC     executat o liniei.
```

Actiunea este de acum foarte diferită de verificarea sintaxei si executia liniei. Dacă sintaxa este verificată atunci forma cu virgulă mobilă trebuie calculată si copiată în linia BASIC actuală. După cum, cînd se execută o linie forma în punct flotant va fi întotdeauna accesibilă asa că este copiată în stiva calulatorului pentru a forma 'ultima valoare'.

```
                CALL  2C9B,DEC-TO-FP        S-a găsit forma în virgulă
                                            mobilă.
                RST   0018,GET-CHAR         Se setează HL pentru a indica
                                            una peste ultima cifră.
                LD    BC,+0006              Sînt cerute sase locatii.
                CALL  1655,MAKE-ROOM        Se face o cameră în linia
                                            BASIC.
                INC   HL                    Se indică primul spatiu liber
                LD    (HL),+0E              Se introduce codul
                                            marcatorului numărului.
                INC   HL                    Se indică a doua locatie.
                EX    DE,HL                 Acest indicator este dorit în
                                            DE.
                LD    HL,(STKEND)           Se aduce STKEND 'vechi'.
                LD    C,+05                 Trebuie mutati 5 octeti.
                AND   A                     Se sterge fanionul de
                                            transport.
                SBC   HL,BC                 STKEND 'nou' = STKEND 'vechi'
                                            -5
                LD    (STKEND),HL           Se mută numărul în virgulă
                LDIR                        mobilă din stiva
```

```
                                                  calculatorului în linie.
                EX    DE,HL                       Se pune indicatorul liniei în
                                                  HL.
                DEC   HL                          Se indică ultimul octet
                                                  adăugat.
                CALL  0077,TEMP-PTR1              Aceasta setează CH-ADD.
                JR    26C3,S-NUMERIC              Salt înainte.
```

In timpul executiei liniei:

```
26B5 S-STK-DEC  RST   0018,GET-CHAR               Se aduce caracterul curent.
26B6 S-SD-SKIP  INC   HL                          Acum se trece la caracterul
                LD    A,(HL)                      următor unul după altul pînă
                CP    +0E                         s-a găsit codul marcatorului
                JR    NZ,26B6,S-SD-SKIP           numărului.
                INC   HL                          Se indică primul octet al
                                                  numărului.
                CALL  33B4,STACK-NUM              Se mută numărul în virgulă
                                                  mobilă.
                LD    (CH-ADD),HL                 Se setează CH-ADD.
```

Acum s-a identificat un rezultat numeric, provenit din RND, PI, ATTR, POINT sau un număr zecimal, de aceea bitul 6 al FLAGS trebuie setat.

```
26C3 S-NUMERIC  SET   6,(FLAGS)                   Se setează fanionul
                JR    26DD,S-CONT-1               marcatorului numeric.
```

THE SCANNING VARIABLE ROUTINE (RUTINA DE BALEIERE VARIABILA)
Cînd s-a identificat numele unei variabile se face un apel la LOOK-VARS care priveste acele variabile care există deja în spatiul variabilelor (sau în spatiul programului, la instructiunile DEF FN pentru o functie definită de utilizator DEF FN). Dacă s-a găsit o valoare numerică corespunzătoare atunci ea este copiată în stiva calculatorului folosind STACK-NUM. Oricum, un sir sau intrarea unei multimi sir trebuie să aibă parametrii corespunzători trecuti în stiva calculatorului de subrutina STK-VAR (sau în cazul unei functii definite de utilizator, de subrutina STK-F-ARG ca si apelată din LOOK-VARS).

```
26C9 S-LETTER   CALL  2882,LOOK-VARS              Se caută printre variabilele
                                                  existente intrarea potrivită.
                JP    C,1C2E,REPORT-2             Se prezintă eroare dacă nu
                                                  este nici o intrare.
                CALL  Z,9996,STK-VARS             Se stochează parametrii
                                                  intrării sirului/returnare
                                                  adresă de bază a elementului
                                                  numeric.
                LD    A,(FLAGS)                   Se aduce FLAGS.
                CP    +C0                         Se testează împreună bitii 6
                                                  si 7.
                JR    C,26DD,S-CONT-1             Unul sau ambii biti sînt
                                                  resetati.
                INC   HL                          Se stochează o valoare
                                                  numerică.
                CALL  33B4,STACK-NUM              Se mută numărul.
26DD S-CONT-1   JR    2712,S-CONT-2               Salt înainte.
```

Caracterul este comparat cu codul pentru '-', care identifică operatia 'minus unar'.
Inainte de testul actual registrul B se setează astfel încît să conțină prioritatea +09 iar registrul C să conțină codul +D8 al operatiei care sînt cerute pentru această operatie.

```
26DF S-NEGATE   LD    BC,+09D8                    Prioritatea +09, codul
                                                  operatiei +D8.
                CP    +2D                         Este un '-'?
                JR    Z,270D,S-PUSH-PO            Salt înainte dacă este un
                                                  'minus unar'.
```

In continuare caracterul este comparat cu codul lui 'VAL$', cu prioritatea 16 zecimal si codul operatiei 18 hex.

```
                LD  BC,+1018                      Prioritatea 16 zecimal, codul
                                                  operatiei +18 hex.
                CP    +AE                         Este 'VALS'?
                JR    Z,270D,S-PUSH-PO            Salt înainte dacă este 'VAL$'
```

Prezentul caracter trebuie să reprezinte acum una dintre functiile CODE la NOT, cu codurile între +AF la +C3.

```
                SUB   +AF                         Intervalul functiei este
```

```
                                          schimbat de la +AF la +C3 la
                                          intervalul +00 la +14 hex.
                JP    C,1C8A,REPORT-C     Se prezintă o eroare dacă se
                                          iasă din interval.
```

Functia 'NOT' este identificată si este tratată separat de celelalte.

```
                LD    BC,+04F0            Prioritatea +04, codul
                                          operatiei +F0.
                CP    +14                 Este functia 'NOT'?
                JR    Z,270D,S-PUSH-PO    Salt dacă este.
                JP    NC,1C8A,REPORT-C    Se verifică din nou
                                          intervalul.
```

Functiile rămase au prioritatea 16 zecimal. Acum se calculează codurile operatiilor pentru aceste functii. Functiile care operează cu siruri necesită ca bitul 6 să fie resetat si functiile care dau rezultate sir necesită bitul 7 resetat în codurile lor de operatii.

```
                LD    B,+10               Prioritatea 16 zecimal.
                ADD   A,+DC               Intervalul functiei este acum
                                          +DC +EF.
                LD    C,A                 Se transferă codul operatiei.
                CP    +DF                 Se separă CODE, VAL si LEN
                JR    NC,2707,S-NO-TO-$   care operează cu siruri
                RES   6,C                 pentru a da rezultate
                                          numerice.
2707 S-NO-TO-$  CP    +EE                 Se separă STR$ si CHR$ care
                JR    C,270D,S-PUSH-PO    operează cu numere si dau
                                          rezultate sir.
                RES   7,C                 Se marchează codul operatiei.
                                          Celelalte coduri de operare
                                          au bitii 6 si 7 setati
                                          amîndoi.
```

Codul prioritătii si codul operatiei pentru functia care a fost considerată este pus acum în stiva masinii. S-a construit astfel o ierarhie a operatiilor.

```
270D S-PUSH-PO  PUSH  BC                  Se stochează codurile de
                RST   0020,NEXT-CHAR      prioritate si al operatiei
                JP    24FF,S-LOOP-1       înainte de a trece la
                                          considerarea următoarei părti
                                          a expresiei.
```

Continuă baleierea liniei. Argumentul prezent poate fi urmat de o '(', un operator binar sau, dacă s-a ajuns la capătul expresiei, atunci un caracter 'carriage return' sau două puncte, un separator sau un 'THEN'.

```
2712 S-CONT-2   RST   0018,GET-CHAR       Se aduce caracterul prezent.
2713 S-CONT-3   CP    +28                 Salt înainte dacă nu este o
                JR    NZ,2723,S-OPERTR    '(' care indică o expresie în
                                          paranteze.
```

Dacă 'ultima valoare' este numerică atunci expresia în paranteze este într-adevăr o subexpresie si trebuie evaluată prin însăsi. După cum, dacă 'ultima valoare' este un sir atunci expresia în parantează reprezintă un element dintr-o multime sau o bucată dintr-un sir. Un apel al SLICING modifică parametrii sirului după cum se cere.

```
                BIT   6,(FLAGS)           Salt înainte dacă se lucrează
                JR    NZ,2734,S-LOOP      cu o expresie numerică în
                                          paranteză.
                CALL  2A52,SLICING        Se modifică parametrii
                                          'ultimei valori'.
                RST   0020,NEXT-CHAR      Se trece la considerarea
                JR    2713,S-CONT-3       următorului caracter.
```

Dacă prezentul caracter este într-adevăr un operator binar va fi dat un cod al operatiei în intervalul +C3 - +CF hex, si un cod de prioritate corespunzător.

```
2723 S-OPERTR   LD    B,+00               Codul initial în BC pentru
                LD    C,A                 indexare în tabelul
                                          operatorilor.
                LD    HL,+2795            Indicatorul în tabel.
                CALL  16DC,INDEXER        Indexul în tabel.
                JR    NC,2734,S-LOOP      Salt înainte dacă nu s-a
                                          găsit nici o operatie.
                LD    C,(HL)              Se aduce codul cerut din
                                          tabel.
```

```
                LD    HL,+26ED          Indicatorul în tabelul de
                                        prioritate: înseamnă că 26ED
                                        +C3 dă 2730 ca prima adresă.
                ADD   HL,BC             Index în tabel.
                LD    B,(HL)            Se aduce prioritatea
                                        corespunzătoare.
```

Acum se introduce bucla principală a acestei subrutine. In această fază există:

i. O 'ultimă valoare' în stiva calculatorului.

ii. Inceputul magaziei de priorități în stiva masinii sub o ierarhie, de mărime necunoscută, a functiilor si codurile de operatie binare. Această ierarhie trebuie invalidată.

iii. Registrul pereche BC contine operatia 'prezentă' si prioritatea, care dacă s-a ajuns la sfîrsitul expresiei va fi prioritate zero.

Initial, 'ultima' operatie si prioritate sînt scoase din stiva masinii si comparate cu 'prezenta' operatie si prioritate.
Dacă prioritatea 'prezentă' este mai mare decît 'ultima' prioritate atunci se iese din buclă, deoarece 'actuala' prioritate este considerată mai prioritară decît 'ultima' prioritate.
Oricum, dacă actuala prioritate este mai mică atunci se efectuează operatia specificată ca 'ultima' operatie. 'Actuala' operatie si prioritate se reîntorc în stiva masinii pentru a reintra în ciclu. In acest fel ierarhia functiilor si operatiile binare care au fost puse la coadă sînt prelucrate în ordine corectă.

```
2734 S-LOOP     POP   DE                Se aduce 'ultima' operatie si
                                        prioritate.
                LD    A,D               Prioritatea trece în
                                        registrul A.
                CP    B                 Se compară 'actuala' cu
                                        'ultima'.
                JR    C,2773,S-TIGHTER  Iesire pentru a astepta
                                        argumentul.
                AND   A                 Sînt ambele priorități zero?
                JP    Z,0018,GET-CHAR   Iesire prin GET-CHAR făcînd
                                        astfel 'ultima valoare'
                                        rezultatul cerut.
```

Inainte de a se executa 'ultima' operatie, se separă functia 'USR' în 'numărul USR' si 'sirul USR' în concordantă cu bitul 6 din FLAGS care a fost setat sau resetat cînd argumentul functiei a fost stocat ca 'ultima valoare'.

```
                PUSH  BC                Se stochează valoarea
                                        'actuală'.
                LD    HL,+5C3B          Adresa FLAGS.
                LD    A,E               'Ultima' operatie este
                CP    +ED               comparată cu codul lui USR,
                JR    NZ,274C,S-STKLST  care va da 'numărul USR' dacă
                                        este modificat; salt dacă nu
                                        este 'USR'.
                BIT   6,(HL)            Se testează bitul 6 al FLAGS.
                JR    NZ,274C,S-STK-LST Salt dacă el este setat
                                        ('număr USR').
                LD    E,+99             Se modifică codul 'ultimei'
                                        operatii: 'deplasament' 19,
                                        +80 pentru intrare sir si
                                        rezultat numeric ('sir USR').
274C S-STK-LST  PUSH  DE                Se stochează 'ultima' valoare
                                        pentru scurt timp.
                CALL  2530,SYNTAX-Z     Nu se efectuează actuala
                JR    Z,275B,Y-SYNTEST  operatie pînă cînd nu s-a
                                        verificat sintaxa.
                LD    A,E               Codul 'ultimei' operatii.
                AND   +3F               Se scot bitii 6 si 7 pentru a
                LD    B,A               converti codul operatiei
                                        într-un deplasament al
                                        calculatorului.
                RST   0028,FP-CALC      Acum se foloseste
                                        calculatorul.
                DEFB  +38,fp-calc-2     Se efectuează operatia
                                        actuală.
                DEFB  +38,end-calc      A fost executată.
                JR    2764,S-RUNTEST    Salt înainte.
```

O pare importantă a verificării sintaxei implică testarea operatiei pentru a se asigura că natura 'ultimi valori' este de tip corect pentru operatie care s-a luat în considerare.

```
2758 S-SYNTEST   LD   A,E              Se aduce codul 'ultimei' operatii.
                 XOR  (FLAGS)         Se compară natura 'ultimei
                 AND  +40             valori' cu cea cerută de
                                      operatie. Trebuie să fie la
                                      fel pentru a fi o sintaxă
                                      corectă.
2761 S-RPORT-C   JP   NZ,1C8A,REPORT-C Salt dacă sintaxa este
                                      gresită.
```

Inainte de a sări înapoi pentru a cicla iar bucla natura 'ultimei valori' trebuie înregistrată în FLAGS.

```
2764 S-RUNTEST   POP  DE              Se aduce codul 'ultimei'
                                      operatii.
                 LD   HL,+5C3B        Adresa FLAGS.
                 SET  6,(HL)          Se presupune că rezultatul
                                      este numeric.
                 BIT  7,E             Salt înainte dacă natura
                 JR   NZ,2770,S-LOOPEND 'ultimei valori' este
                                      numerică.
                 RES  6,(HL)          Este sir.
2770 S-LOOPEND   POP  BC              Se aduce valoarea 'prezentă'
                                      în BC.
                 JR   2734,S-LOOP     Salt înapoi.
```

De cîte ori operatia 'actuală' este mai prioritară, 'ultima' si 'actuala' valoare se întorc în stiva masinii. Oricum, dacă operatia 'actuală' cere un sir ca operand atunci codul operatiei este modificat pentru a indica această cerintă.

```
2773 S-TIGHTER   PUSH DE              'Ultima' valoare trece în
                                      stivă.
                 LD   A,C             Se aduce codul operatiei
                                      'actuale'.
                 BIT  6,(FLAGS)       Nu se modifică codul
                 JR   NZ,2790,S-NEXT  operatiei dacă se lucrează cu
                                      un operand numeric.
                 AND  +3F             Se sterg bitii 6 si 7.
                 ADD  A,+08           Se incrementează codul cu +08
                                      hex.
                 LD   C,A             Se returnează codul în
                                      registrul C.
                 CP   +10             Este operatia 'AND'?
                 JR   NZ,2788,S-NOT-AND Salt dacă nu este asa.
                 SET  6,C             'AND' cere un operand numeric
                 JR   2790,S-NEXT     Salt înainte.
2788 S-NOT-AND   JR   C,2761,S-RPORT-C Operatiile -,*,/, si OR nu
                                      sînt posibile între siruri.
                 CP   +17             Este operatia '+'?
                 JR   Z,2790,S-NEXT   Salt dacă este asa.
                 SET  7,C             Celelalte operatii produc un
                                      rezultat numeric.
2790 S-NEXT      PUSH BC              Valoarea 'actuală' trece în
                                      stiva masinii.
                 RST  0020,NEXT-CHAR  Se consideră următorul
                                      caracter.
                 JP   24FF,S-LOOP-1   Se ciclează din nou bucla.
```

THE TABLE OF OPERATORS (TABELUL OPERATORILOR)

locatie	cod operator	operator	locatie	cod operator	operator
2795	2B	+	27A3	3C	CD	<
2797	2D	-	27A5	C7	C9	<=
2799	2A	*	27A7	C8	CA	>=
289B	2F	/	27A9	C9	CB	<>
279D	5E	↑	27AB	C5	C7	OR
279F	3D	=	27AD	C6	C8	AND
27A1	3E	>	27AF	00	End marker	

THE TABLE OF PRIOTITIES (TABELUL PRIORITATILOR)

locatie	prioritate	operator	locatie	prioritate	operator

27B0	06	-	27B7	05	>=
27B1	08	*	27B8	05	<>
27B2	08	/	27B9	05	>
27B3	0A		27BA	05	<
27B4	02	OR	27BB	05	=
27B5	03	AND	27BC	06	+
27B6	05	<=			

THE 'SCANNING FUNCTION' SUBROUTINE (SUBRUTINA 'BALEIERE FUNCTIE')
Această subrutină este apelată de 'rutina de baleiere FN' pentru a evalua o funcție definită de utilizator care intervine într-o linie BASIC. Subrutina poate fi considerată în patru faze:

i. Sintaxa instrucției FN este verificată în timpul verificării sintaxei.

ii. In timpul execuției liniei, se efectuează o căutare în spațiul programului pentru o instrucțiune DEF FN, și numele funcțiilor sînt comparate pînă se găsește o egalitate - sau se raportează o eroare.

iii. Argumentele lui FN sînt evaluate prin apelarea lui SCANNING.

iv. Funcția este evaluată ea însăși prin apelarea lui SCANNING, care apelează pe rînd LOOK-VARS și astfel subrutina 'STACK FUNCTION ARGUMENT'.

```
27BD S-FN-SBRN    CALL  2530,SYNTAX-Z       Se face un salt la SF-RUN
                  JR    NZ,27F7,SF-RUN      numai dacă s-a verificat
                                            sintaxa.
                  RST   0020,NEXT-CHAR      Se aduce primul caracter al
                                            numelui.
                  CALL  2C3D,ALPHA          Dacă nu este alfabetic, se
                  JP    NC,1C8A,REPORT-C    prezintă eroare.
                  RST   0020,NEXT-CHAR      Se aduce următorul caracter.
                  CP    +24                 Este '$'?
                  PUSH  AF                  Se salvează fanionul zero în
                                            stivă.
                  JR    NZ,27D0,SF-BRKT-1   Salt dacă nu a fost '$'.
                  RST   0020,NEXT-CHAR      Se aduce următorul caracter.
27D0 SF-BRKT-1    CP    +28                 Dacă nu este un caracter '(',
                  JR    NZ,27E6,SF-RPRT-C   se prezintă eroare.
                  RST   0020,NEXT-CHAR      Se aduce următorul caracter.
                  CP    +29                 Este ')'?
                  JR    Z,27E9,SF-FLAG-6    Salt dacă este; nu mai există
                                            argumente.
27D9 SF-ARGMTS    CALL  24FB,SCANNING       In buclă se apelează SCANNING
                                            pentru a verifica sintaxa
                                            fiecărui argument și pentru a
                                            introduce numere în virgulă
                                            mobilă.
                  RST   0018,GET-CHAR       Se aduce caracterul care
                  CP    +2C                 urmează după argument; salt
                  JR    NZ,27E4,SF-BRKT-2   dacă nu este ',' - nu mai
                                            sînt argumente.
                  RST   0020,NEXT-CHAR      Se aduce primul caracter din
                                            următorul argument.
                  JR    27D9,SF-ARGMTS      Salt înapoi pentru a lua în
                                            considerare acest argument.
27E4 SF-BRKT-2    CP    +29                 Este caracterul curent ')'?
27E6 SF-RPRT-6    JP    NZ,1C8A,REPORT-C    Se prezintă eroare dacă nu
                                            este.
27E9 SF-FLSG-6    RST   0020,NEXT-CHAR      Se indică următorul caracter
                                            din linia BASIC.
                  LD    HL,+5C3B            FLAGS; se presupune o funcție
                  RES   6,(HL)              valoare sir și se resetează
                                            bitul 6 al lui FLAGS.
                  POP   AF                  Se reface fanionul zero; salt
                  JR    Z,27F4,SF-SYN-EN    dacă FN este într-adevăr o
                                            valoare sir.
                  SET   6,(HL)              Altfel, se setează bitul 6 al
                                            lui FLAGS.
27F4 SF-SYN-EN    JP    2712,S-CONT-2       Salt înapoi pentru a continua
                                            baleierea liniei.
```

ii. In timpul execuției liniei, mai întîi trebuie executată o căutare pentru o instrucțiune DEF FN.

```
27F7 SF-RUN       RST   0020,NEXT-CHAR      Se aduce primul caracter al
                                            numelui.
                  AND   +DF                 Resetare bit 5.
```

```
                LD    B,A           Se copiază numele în B.
                RST   0020,NEXT-CHAR Se aduce următorul caracter.
                SUB   +24           Se scade 24 hex, codul pentru
                                    '$'.
                LD    C,A           Se copiază rezultatul în C
                                    (zero pentru un sir, non-zero
                                    pentru o functie numerică).
                JR    NZ,2802,SF-ARGMT1 Salt dacă non-zero: functie
                                    numerică.
                RST   0020,NEXT-CHAR Se aduce următorul caracter,
                                    '('.
2802 SF-ARGMT1  RST   0020,NEXT-CHAR Se aduce primul caracter al
                                    primului argument.
                PUSH  HL            Se salvează indicatorul lui
                                    în stivă.
                LD    HL,(PROG)     Se indică începutul
                                    programului.
                DEC   HL            Întoarcere cu o locatie.
2808 SF-FND-DF  LD    DE,+00CE      Se caută 'DEF FN'.
                PUSH  BC            Salvare nume si 'stare sir'.
                CALL  1D86,LOOK-PROG Se caută acum programul.
                POP   BC            Se readuce numele si starea.
                JR    NC,2814,SF-CP-DEF Salt dacă s-a găsit o
                                    instructie DEF FN.

Prezentarea P - FN fără DEF

2812 REPORT-P   RST   0008,ERROR-1  Se apelează rutina de tratare
                DEFB  +18           eroare.
```

Cînd s-a găsit o instructiune DEF FN se comparaă numele si starea celor două functii; dacă ele nu coincid, se reia căutarea.

```
2814 SF-CP-DEF  PUSH  HL            Se salvează indicatorul
                                    caracterului DEF FN în cazul
                                    în care căutarea trebuie
                                    reluată.
                CALL  28AB,FN-SKPOVR Se aduce numele functiei DEF
                                    FN.
                AND   +DF           Resetare bit 5.
                CP    B             Coincide cu numele FN?
                JR    NZ,2825,SF-NOT-FD Salt dacă nu a coincis.
                CALL  28AB,FN-SKPOVR Se aduce următorul caracter
                                    în DEF FN.
                SUB   +24           Se scade 24 hex, codul pentru
                                    '$.
                CP    C             Se compară starea cu cea din
                                    FN.
                JR    Z,2831,SF-VALUES Salt dacă s-a găsit
                                    coincidentă .
2825 SF-NOT-FD  POP   HL            Se repune indicatorul pe 'DEF
                                    FN'.
                DEC   HL            Reîntoarcere o locatie.
                LD    DE,+0200      Se foloseste rutina de
                PUSH  BC            căutare pentru a găsi
                CALL  198B,EACH-STMT sfîrsitul instructiei DEF FN,
                POP   BC            pregătind următoarea căutare;
                                    în acelasi timp se salvează
                                    numele si starea.
                JR    2808,SF-FND-DF Salt înapoi pentru o căutare
                                    ulterioară.
```

iii. Acum s-a găsit instructiunea DEF FN corectă. Argumentele instructiunii FN vor fi evaluate prin apelarea repetată a lui SCANNING, si cei 5 octeti valoare (sau parametrii, pentru sir) vor fi instrodusi în instructiunea DEF FN în spatiile rezervate la verificarea sintaxei. HL va fi folosit pentru a indica de-a lungul instructiunii DEF FN (apelînd FN-SKPOVR cînd este necesar) în timp ce CH-ADD indică de-a lungul instructiunii FN (apelînd RST 0020, NEXT-CHAR, cînd este necesar).

```
2831 SF-VALUES  AND   A             Dacă acum HL indiă un '$', se
                CALL  Z,28AB,FN-SKPOVR trece la '('.
                POP   DE            Se înlătură indicatorul lui
                                    'DEF FN'.
                POP   DE            Se trece indicatorul la
                LD    (CH-ADD),DE   primul argument al lui FN, si
                                    se copiază în CH-ADD.
                CALL  28AB,FN-SKPOVR Acum se trece peste '('.
                PUSH  HL            Se salvează acest indicator
```

```
                        CP    +29          în stivă.
                        JR    Z,2885,SF-R-BR-2   Indică acesta ')'?
                                           Dacă da, salt: FN nu are
                                           argumente.
2843 SF-ARG-LP          INC   HL           Se indică următorul cod.
                        LD    A,(HL)       Se pune codul în A.
                        CP    +0E          Este codul 'marcator număr'
                                           0E hex?
                        LD    D,+40        Pentru un argument numeric se
                                           setează bitul 6 al lui D.
                        JR    Z,2852,SF-ARG-VL   Salt la zero: argument
                                           numeric.
                        DEC   HL           Acum se asigură că HL indică
                        CALL  28AB,FN-SKPOVR caracterul '$' (nu un cod de
                                           control).
                        INC   HL           HL indică acum 'marcatorul
                                           numărului'.
                        LD    D,+00        Bitul 6 a lui D este resetat:
                                           argument sir.
2852 SF-ARG-VL          INC   HL           Se indică primul din cei
                                           cinci octeti din DEF FN.
                        PUSH  HL           Se salvează acest indicator
                                           în stivă.
                        PUSH  DE           Se salvează 'starea sir' a
                                           argumentului.
                        CALL  24FB,SCANNING Acum se evaluează argumentul.
                        POP   AF           Se pune fanionul nu/sir în A.
                        XOR   (FLAGS)      Se compară bitul 6 cu
                        AND   +40          rezultatul din SCANNING.
                        JR    NZ,288B,REPORT-Q Se dă prezentarea Q dacă nu
                                           coincid.
                        POP   HL           Se aduce indicatorul primului
                        EX    DE,HL        din cele 5 spatii din DEF FN
                                           în DE.
                        LD    HL,(STKEND)  Indică HL la STKEND.
                        LD    BC,+0005     BC va număra 5 octeti care
                                           trebuie mutati.
                        SBC   HL,BC        Mai întîi se decrementează
                        LD    (STKEND),HL  cu 5, asa că se sterge
                                           'ultima valoare' din stivă.
                        LDIR               Se copiază octetii în spatiul
                                           din DEF FN.
                        EX    DE,HL        HL indică următorul cod.
                        DEC   HL           Se asigură că HL indică
                        CALL  28AB,FN-SKPOVR caracterul de după cei 5
                                           octeti.
                        CP    +29          Este ')'?
                        JR    Z,2885,SF-R-BR-2   Salt dacă este: nu mai sînt
                                           argumente în instructiunea
                                           DEF FN.
                        PUSH  HL           Este ',': se salvează
                                           indicatorul în el.
                        RST   0018,GET-CHAR Se aduce caracterul de după
                                           ultimul argument din FN care
                                           a fost evaluat.
                        CP    +2C          Salt dacă nu este ',': nu
                        JR    NZ,288B,REPORT-Q este coincidentă între
                                           argumentele din FN si DEF FN.
                        RST   0020,NEXT-CHAR Se pune CH-ADD pe următorul
                                           argument al lui FN.
                        POP   HL           HL indică din nou ',' din DEF
                                           FN.
                        CALL  28AB,FN-SKPOVR Se mută HL la următorul
                                           argument din DEF FN.
                        JR    2843,SF-ARG-LP Salt înapoi pentru a lua în
                                           considerare acest argument.
2885 SF-R-BR-2          PUSH  HL           Se salvează indicatorul lui
                                           ')' în DEF FN.
                        RST   0018,GET-CHAR Se aduce caracterul de după
                                           ultimul argument din FN.
                        CP    +29          Este ')'?
                        JR    Z,288D,SF-VALUE Dacă da, salt pentru a evalua
                                           functia; dacă nu, se dă
                                           prezentarea Q.
```

REPORT Q - Eroare parametru

```
288B REPORT-Q           RST   0008,ERROR-1 Se apelează rutina de tratare
                        DEFB  +19          eroare.
```

iv. In final, functia este ea însăsi evaluată prin apelarea lui SCANNING, după ce mai întîi s-a setat DEFADD pentru a conține adresa argumentelor așa cum intervin în instructiunea DEF FN. Aceasta asigură că LOOK-VARS, cînd apelează SCANNING, va căuta mai întîi aceste argumente pentru valorile cerute, înainte de o căutare în spatiul variabilelor.

```
288D SF-VALUE   POP   DE              Readucere indicator pe ')' în
                                       DEF FN.
                EX    DE,HL            Se aduce acest indicator în
                                       HL.
                LD    (CH-ADD),HL      Se introduce în CH-ADD.
                LD    HL,(DEFADD)      Se aduce vechea valoare a lui
                                       DEFADD.
                EX    (SP),HL          Ea se stochează, si se aduce
                LD    (DEFADD),HL      adresa de început a spatiului
                                       argumentelor din DEF FN în
                                       DEFADD.
                PUSH  DE               Se salvează adresa lui ')' în
                                       FN.
                RST   0020,NEXT-CHAR   Se mută CH-ADD trecînd peste
                RST   0020,NEXT-CHAR   ')' si '=' la începutul
                                       expresiei pentru functia din
                                       DEF FN.
                CALL  24FB,SCANNING    Acum se evaluează functia.
                POP   HL               Se reface adresa lui ')' în
                                       FN.
                LD    (CH-ADD),HL      Ea se salvează în CH-ADD.
                POP   HL               Se readuce valoarea initială
                                       a lui DEFADD.
                LD    (DEFADD),HL      Se pune înapoi în DEFADD.
                RST   0020,NEXT-CHAR   Se aduce următorul caracter
                                       din linia BASIC.
                JP    2712,S-CONT-2    Salt înapoi pentru a continua
                                       baleierea.
```

THE 'FUNCTION SKIPOVER' SUBROUTINE
Această subrutină este folosită de FN si de STK-F-ARG pentru a-l muta pe HL de-a lungul instructiunii DEF FN lăsînd CH-ADD neschimbat, cînd el indică de-a lungul instructiunii FN.

```
28AB FN-SKPOVR  INC   HL               Se indică următorul cod din
                                       instructie.
                LD    A,(HL)           Se copiază codul în A.
                CP    +21              Salt înapoi pentru a sări
                JR    C,28AB,FN-SKPOVR peste el dacă este un cod
                                       control sau un spatiu.
                RET                    Sfîrsit.
```

THE 'LOOK-VARS' SUBROUTINE
Această subrutină este apelată întotdeauna cînd se cere o căutare în spatiul variabilelor sau al argumentelor unei instructiuni DEF FN. Subrutina este introdusă cu variabila sistem CH-ADD indicînd prima literă a numelui variabilei a cărei locatie a fost căutată. Numele va fi în spatiul programului sau în zona de lucru. Initial subrutina construieste un octet discriminator, în registrul C, care este bazat pe prima literă a numelui variabilei. Bitii 5 & 6 indică tipul variabilei care a fost tratată.
Registrul B este folosit ca un registru bit care contine fanioane.

```
28B2 LOOK-VARS  SET   6,(FLAGS)        Se presupune o variabilă
                                       numerică.
                RST   0018,GET-CHAR    Se aduce primul caracter în A
                CALL  2C8D,ALPHA       Este alfabetic?
                JP    NC,1C8A,REPORT-C Se prezintă eroare dacă nu
                                       este așa.
                PUSH  HL               Se salvează indicatorul
                                       pentru prima literă.
                AND   +1F              Se transferă bitii 0 la 4 ai
                LD    C,A              literei în registrul C; bitii
                                       5 & 7 sînt întotdeauna
                                       resetati.
                RST   0020,NEXT-CHAR   Se aduce al 2-lea caracter în
                                       A.
                PUSH  HL               Si acest indicator este
                                       salvat.
                CP    +28              Este al 2-lea caracter '('?
                JR    Z,28EF,V-RUN/SYN Se separă multimea numerelor.
                SET   6,C              Acum se setează bitul 6.
                CP    +24              Este al 2-lea caracter '$'?
```

```
                JR    Z,28DE,V-STR-VAR      Se separă toate sirurile.
                SET   5,C                   Acum se setează bitul 5.
                CALL  2C88,ALPHANUM         Salt în față dacă numele
                JR    NC,28E3,V-TEST-FN     variabilei are doar un
                                            caracter.
```

Acum se găseste caracterul final al unui nume care are mai mult de un caracter.

```
28D4 V-CHAR     CALL  2C88,ALPHANUM         Este caracter alfanumeric?
                JR    NC,28EF,V-RUN/SYN     Salt afară din buclă cînd s-a
                                            găsit sfîrsitul numelui.
                RES   6,C                   Se marchează octetul
                                            discriminator.
                RST   0020,NEXT-CHAR        Salt înapoi pentru a-l testa.
```

Sirurile simple si multimile de siruri cer ca bitul 6 al lui FLAGS să fie resetat.

```
28DE V-STR-VAR  RST   0020,NEXT-CHAR        CH-ADD trece peste '$'.
                RES   6,(FLAGS)             Se resetează bitul 6 pentru a
                                            indica un sir.
```

Dacă DEFADD este non-zero, indicînd că o 'functie' (o 'FN') a fost evaluată, si dacă este întimpul executiei', se execută o căutare a argumentelor din instructia DEF FN.

```
28E3 V-TEST-FN  LD    A,(DEFADD-hi)         DEFADD-hi este zero?
                AND   A
                JR    Z,28EF,V-RUN/SYN      Dacă da, salt înainte.
                CALL  2530,SYNTAX-Z         Este în 'timpul executiei'?
                JP    NZ,2951,STK-F-ARG     Dacă da, salt înainte pentr
                                            a găsi instructiunea DEF FN.
```

In caz contrar (sau dacă variabila nu a fost găsită în instructiunea DEF FN) se execută căutarea în spatiul variabilelor numai dacă sintaxa a fost verificată.

```
28EF V-RUN/SYN. LD    B,C                   Se copiază octetul
                                            discriminator în registrul B.
                CALL  2530,SYNTAX-Z         Salt înainte dacă este în
                JR    NZ,28FD,V-RUN         'timpul executiei'.
                LD    A,C                   Se mută discriminatorul în A.
                AND   +E0                   Se coboară partea de cod
                                            caracter.
                SET   7,A                   Se indică sintaxa setînd
                                            bitul 7.
                LD    C,A                   Se readuce discriminatorul.
                JR    2934,V-SYNTAX         Salt înainte pentru a
                                            continua.
```

S-a executat o linie BASIC asa că se execută o căutare în spatiul variabilelor.

```
28FD V-RUN      LD    HL,(VARS)             Se scoate indicatorul VARS.
```

Acum se introduce o buclă pentru a lua în considerare numele variabilelor existente.

```
2900 V-EACH     LD    A,(HL)                Prima literă a fiecărei
                                            variabile existente.
                AND   +7F                   Egalare biti 0 la 6.
                JR    Z,2932,V-80-BYTE      Salt cînd se ajunge la
                                            'octetul 80'.
                CP    C                     Comparare actuală.
                JR    NZ,292A,V-NEXT        Salt înainte dacă primele
                                            caractere nu coincid.
                RLA                         Se roteste A la stînga si
                ADD   A,A                   apoi se dublează pentru
                                            testarea bitilor 5 & 6.
                JP    P,293F,V-FOUND-2      Siruri si multime de
                                            variabile.
                JR    C,293F,V-FOUND-2      Numeric simplu si variabile
                                            FOR-NEXT.
```

Se cere ca numele întregi să coincidă în întregime.

```
                POP   DE                    Se face o copie a
                PUSH  DE                    indicatorului pentru cel de
```

```
                                                    al doilea caracter.
                PUSH  HL                            Se salvează indicatorul
                                                    primei litere.
2912 V-MATCHES  INC   HL                            Se consideră următorul
                                                    caracter.
2913 V-SPACES   LD    A,(DE)                        Se aduce pe rînd fiecare
                                                    caracter.
                INC   DE                            Se indică următorul caracter.
                CP    +20                           Este caracterul 'spatiu'?
                JR    Z,2913,V-SPACES               Se ignoră spatiile.
                OR    +20                           Se setează bitul 5 astfel
                                                    încît să se egaleze portiunea
                                                    literelor mici si mari.
                CP    (HL)                          Se face compararea.
                JR    Z,2912,V-MATCHES              Intoarcere pentru un alt
                                                    caracter în caz de egalitate.
                OR    +80                           Va coincide cu bitul 7 setat?
                CP    (HL)                          Se încearcă.
                JR    NZ,2929,V-GET-PTR             Salt înainte dacă 'ultimele
                                                    caractere' nu coincid.
                LD    A,(DE)                        Se verifică dacă s-a ajuns la
                CALL  2C88,ALPHANUM                 sfîrsitul numelui înainte de
                JR    NC,293E,V-FOUND-1             a executa salt înainte.
```

In toate cazurile în care numele nu coincid registrul pereche HL trebuie făcut să indice următoarea variabilă din spatiul variabilelor.

```
2929 V-GET-PTR  POP   HL                            Se aduce indicatorul.
292A V-NEXT     PUSH  BC                            Se salvează B si C pentru
                                                    scurt timp.
                CALL  19B8,NEXT-ONE                 DE este făcut să indice
                                                    următoarea variabilă.
                EX    DE,HL                         Se schimbă cei doi indicatori
                POP   BC                            Se aduc B & C înapoi.
                JR    2900,V-EACH                   Se intră din nou în buclă.
```

Se revine în acest punct dacă nu s-a găsit nici o intrare cu numele corect.

```
2932 V-80-BYTE  SET   7,B                           Semnalizare 'nu s-a găsit
                                                    variabila'.
```

Se vine în acest punct dacă se varifică sintaxa.

```
2934 V-SYNTAX   POP   DE                            Se coboară indicatorul pe al
                                                    doilea caracter.
                RST   0018,GET-CHAR                 Se aduce caracterul prezent.
                CP    +28                           Este '('?
                JR    Z,2943,V-PASS                 Salt înainte.
                SET   5,B                           Se indică faptul că nu se
                JR    294B,V-END                    lucrează cu o multime si salt
                                                    înainte.
```

Se vine în acest punct dacă s-a găsit o intrare cu numele corect.

```
293E V-FOUND-1  POP   DE                            Se coboară indicatorul
                                                    variabilei salvate.
293F V-FOUND-2  POP   DE                            Se coboară indicatorul celui
                                                    de al doilea caracter.
                POP   DE                            Se coboară indicatorul primei
                                                    litere.
                PUSH  HL                            Se salvează indicatorul
                                                    'ultimei' litere.
                RST   0018,GET-CHAR                 Se aduce caracterul curent.
```

Dacă numele variabilei egalate are mai mult de o literă atunci se trece peste celelalte caractere.

Notă: Aceasta pare că s-ar fi făcut deja la V-CHAR.

```
2943 V-PASS     CALL  2C,88,ALPHANUM                Este alfanumeric?
                JR    NC,294B,V-END                 Salt cînd s-a găsit sfîrsitul
                                                    numelui.
                RST   0020,NEXT-CHAR                Se aduce următorul caracter.
                JR    2943,V-PASS                   Intoarcere pentru a-l testa.
```

Acum se setează parametrii de iesire.

```
294B V-END      POP   HL                            HL contine indicatorul
                                                    literei unui nume scurt sau
```

```
                                        'ultimul' caracter al unui
                                        nume lung.
        RL    B                         Se roteste întregul registru.
        BIT   6,B                       Se specifică starea bitului 6
        RET                             Sfîrsit.
```

Parametrii de iesire ai subrutinei pot fi prezentati după cum urmează: Variabila sistem CH-ADD indică prima locatie de după numele variabilei asa cum intervine în linia BASIC.

Dacă 'nu se găseste variabila':

i. Fanionul de transport este setat.

ii. Fanionul zero este setat numai cînd căutarea s-a făcut pentru o multime variabilă.

iii. Registrul pereche HL indică prima literă a numelui variabilei asa cum intervine într-o linie BASIC.

Dacă 's-a găsit variabila':

i. Fanionul de transport este resetat.

ii. Fanionul zero este setat pentru ambele siruri simple de variabile si toate variabilele multimii.

iii. Registrul pereche HL indică litera unui nume 'scurt', sau 'ultimul' caracter al unui nume lung, sau intrarea existentă care s-a găsit în spatiul variabilelor.

In toate cazurile bitii 5 & 6 ai registrului C indică tipul variabilei care a fost tratată. Bitul 7 este complementul fanionului SYNTAX/RUN. Numai cînd subrutina este folosită în 'timpul executiei' bitii 0 la 4 vor contine codul literei variabilei.

In timpul sintaxei revenirea se face întotdeauna cu fanionul de transport resetat. Fanionul zero este setat pentru multimi si resetat pentru toate celelalte variabile, exceptînd un nume sir simplu incorect urmat de o '(' care setează fanionul zero si, în cazul SAVE "nume" DATA un $(), de asemenea trece sintaxa ???

THE 'STACK FUNCTION ARGUMENT' SUBROUTINE (SUBRUTINA 'STOCARE ARGUMENT FUNCTIE')
Această subrutină este apelată de LOOK-VARS cînd DEFADD-hi nu este zero, pentru a efectua o căutare în spatiul argumentelor instructiunii DEF FN, înainte de a căuta în spatiul variabilelor. Dacă variabila s-a găsit în intructiunea DEF FN, atunci parametrii unei variabile sir sînt stocati si se dă un semnal că nu este necesar să se apeleze STK/VAR. Dar se permite baleierea în stivă a valorii unei variabile numerice la 26DA în mod uzual.

```
2951 STK-ARG    LD    HL,(DEFADD)       Se indică primul caracter din
                LD    A,(HL)            spatiul argumentelor si se
                                        pune în A.
                CP    +29               Este ')'?
                JP    Z,28EF,V-RUN/SYN  Salt pentru a cerceta spatiul
                                        variabilelor.
295A SFA-LOOP   LD    A,(HL)            Se aduce următorul argument
                                        în buclă.
                OR    +60               Se setează bitii 5 & 6,
                LD    B,A               initializînd o variabilă
                                        numerică simplă; aceasta se
                                        copiază în B.
                INC   HL                Se indică următorul cod.
                LD    A,(HL)            Acesta se pune în registrul A
                CP    +0E               Este codul 'marcatorului
                                        numărului' 0E hex.
                JR    Z,296B,SFA-CP-VR  Salt dacă este așa: variabilă
                                        numerică.
                DEC   HL                Asigurare că HL indică un
                CALL  28AB,FN-SKPOVR    caracter '$', si nu un cod
                                        control sau săptiu.
                INC   HL                HL indică acum 'marcatorul
                                        numărului'.
                RES   5,B               Se resetează bitul 5 al lui
                                        B: variabilă sir.
296B SFA-CP-VR  LD    A,B               Se aduce numele variabilei în
                                        A.
                CP    C                 Este una din cele căutate?
```

```
                JR    Z,2981,SFA-MATCH      Salt dacă este.
                INC   HL                    Acum se trece peste cei cinci
                INC   HL                    octeti ai numărului în
                INC   HL                    virgulă flotantă sau
                INC   HL                    parametrii sir pentru a
                INC   HL                    ajunge la următorul argument.
                CALL  28AB,FN-SKPOVR        Se trece la următorul
                                            caracter.
                CP    +29                   Este ')'?
                JP    Z,28EF,V-RUN/SYN      Dacă este, salt pentru a
                                            cerceta spatiul variabilelor.
                CALL  28AB,FN-SKPOVR        Se indică următorul argument.
                JR    295A,SFA-LOOP         Salt înapoi pentru a-l lua în
                                            considerare.
```

S-a găsit o egalitate. Parametrii unei variabile sir sînt stocati, evitînd necesitatea apelării subrutinei STK-VAR.

```
2981 SFA-MATCH  BIT   5,C                   Se testează dacă este o
                                            variabilă numerică.
                JR    NZ,2991,SFA-END       Salt dacă variabila este
                                            numerică; va fi stocată de
                                            SCANNING.
                INC   HL                    Se indică primul din cei 5
                                            octeti ce trebuie stocati.
                LD    DE,(STKEND)           In DE se indică STKEND.
                CALL  33C0,MOVE-FP          Se stochează cei 5 octeti.
                EX    DE,HL                 Se indică în HL noua pozitie
                LD    (STKEND),HL           a lui STKEND, si se resetează
                                            variabila sistem.
2991 SFA-END    POP   DE                    Se scot indicatorii LOOK-VARS
                POP   DE                    (al doilea si primul
                                            indicator caracter).
                XOR   A                     Se revine din căutare cu
                INC   A                     fanioanele de transport si
                                            zero resetate - semnalizînd
                                            că nu se cere o apelare a lui
                                            STK-VAR.
                RET                         Sfîrsit.
```

THE 'STK-VAR' SUBROUTINE

Această subrutină se foloseste de obicei fie pentru a găsi parametrii care definesc intrarea unui sir existent în spatiul variabilelor, fie pentru a returna registrului pereche HL adresa de bază a unui element particular sau a unei multimi de numere. Cînd apelul se face din DIM subrutina doar verifică sintaxa instructiei BASIC.

De notat că parametrii care definesc un sir pot fi modificati apelînd SLICING dacă acesta poate fi specificată.

Initial registrii A si B sînt stersi si bitul 7 al registrului C este testat pentru a determina dacă s-a verificat sintaxa.

```
2996 STK-VAR    XOR   A                     Se sterge dispunerea
                                            fanioanelor.
                LD    B,A                   Se sterge registrul B pentru
                                            mai tîrziu.
                BIT   7,C                   Salt înainte dacă s-a
                JR    NZ,29E7,SV-COUNT      verificat sintaxa.
```

In continuare sirurile simple sînt separate de multimea variabilelor.

```
                BIT   7,(HL)                Salt înainte dacă se lucrează
                JR    NZ,29AE,SV-ARRAYS     cu o multime de variabile.
```

Parametrii unui sir simplu s-au găsit usor.

```
                INC   A                     Semnalizare 'un sir simplu'.
29A1 SV-SIMPLE$ INC   HL                    Trece de-a lungul intrării.
                LD    C,(HL)                Se scoate partea inferioară a
                                            contorului.
                INC   HL                    Avans indicator.
                LD    B,(HL)                Se scoate partea superioară a
                                            contorului.
                INC   HL                    Avans indicator.
                EX    DE,HL                 Transfer indicator în sirul
                                            actual.
                CALL  2AB2,STK-STORE        Se trec acesti parametrii în
                                            stiv calculatorului.
                RST   0018,GET-CHAR         Se aduce caracterul prezent
                                            si se execută salt înainte
```

```
                JP    2A49,SV-SLICE          pentru a vedea dacă se cere o
                                             'tăiere'.
```

Acum se găseste adresa de bază a unui element dintr-o multime. Initial se colectează 'numărul dimensiunilor'.

```
29AE SV-ARRAYS  INC   HL                     Se trece peste octetii
                INC   HL                     lungime.
                INC   HL
                LD    B,(HL)                 Se colectează 'numărul
                                             dimensiunilor'.
                BIT   6,C                    Salt înainte dacă se trateaza
                JR    Z,29C0,SV-PTR          o multime de numere.
```

Dacă o multime de siruri are 'numărul său de dimensiuni' egal cu '1' atunci o astfel de multime poate fi tratată ca un sir simplu.

```
                DEC   B                      Se decrementează 'numărul
                JR    Z,29A1,SV-SIMPLE$      dimensiunilor' si se execută
                                             salt dacă acum acest număr
                                             este zero.
```

In continuare se face o verificare pentru a se asigura că în linia BASIC variabila este urmată de o semnătură.

```
                EX    DE,HL                  Salvare indicator în DE.
                RST   0018,GET-CHAR          Se aduce caracterul prezent.
                CP    +28                    Este '('?
                R     NZ,2A20,REPORT-3       Se prezintă eroare dacă nu
                                             este asa.
                EX    DE,HL                  Se reface indicatorul.
```

Si pentru multimile numerice si pentru multimile de siruri indicatorul variabilei este transferat în registrul pereche DE înainte de evaluarea semnăturii.

```
29C0 SV-PTR     EX    DE,HL                  Se trece indicatorul în DE.
                JR    29E7,SV-COUNT          Salt înainte.
```

Următoarea buclă este folosită pentru a găsi parametrii unui element specificat în cadrul unei multimi. Bucla este introdusă prin punctul de mijloc - SV-COUNT - unde contorul elementelor este setat pe zero.
Bucla este accesată de 'B' ori, aceasta fiind, pentru o multime numerică, egal cu numărul dimensiunilor care au fost folosite, dar pentru o multime de siruri 'B' este cu unu mai mic decît numărul dimensiunilor în folosintă cum ultima semnătură este folosită pentru a specifica 'segment' al sirului.

```
29C3 SV-COMMA   PUSH  HL                     Salvare contor.
                RST   0018,GET-CHAR          Se aduce caracterul prezent.
                POP   HL                     Se reface contorul.
                CP    +2C                    Este carterul prezent o ')'?
                JR    Z,29EA,SV-LOOP         Salt înainte pentru a
                                             considera o altă semnătură.
                BIT   7,C                    Dacă o linie a fost executată
                JR    Z,2A20,REPORT-3        atunci acolo este o eroare.
                BIT   6,C                    Salt înainte dacă se lucrează
                JR    NZ,29D8,SV-CLOSE       cu o multime de siruri.
                CP    +29                    Este caracterul prezent ')'?
                JR    NZ,2A12,SV-RPT-C       Se dă eroare dacă nu este asa
                RST   0020,NEXT-CHAR         Avans CH-ADD.
                RET                          Revenire dacă sintaxa este
                                             corectă.
```

Pentru o multime de siruri semnătura prezentă poate reprezenta un 'segment', sau semnătura pentru un 'segment' poate fi încă prezentă în linia BASIC.

```
29D8 SV-CLOSE   CP    +29                    Este caracterul prezent ')'?
                JR    Z,2A48,SV-DIM          Salt înainte si verificare
                                             dacă acolo mai este o altă
                                             semnătură.
                CP    +CC                    Este caracterul prezent 'TO'?
                JR    NZ,2A12,SV-RPT-C       Nu trebuie să fie altfel.
29E0 SV-CH-ADD  RST   0018,GET-CHAR          Se aduce caracterul prezent.
                DEC   HL                     Se indică precedentul
                LD    (CH-ADD),HL            caracter si se setează CH-ADD
                JR    2A45,SV-SLICE          Evaluarea 'segment'.
```

Introducere buclă aici.

```
29E7 SV-COUNT   LD    HL,+0000          Setare contor pe zero.
29EA SV-LOOP    PUSH  HL                Se salvează contorul pentru
                                        scurt timp.
                RST   0020,NEXT-CHAR    Avans CH-ADD.
                POP   HL                Se reface contorul.
                LD    A,C               Se aduce octetul
                                        discriminator.
                CP    +C0               Salt numai dacă s-a verificat
                JR    NZ,29FB,SV-MULT   sintaxa pentru o multime de
                                        siruri.
                RST   0018,GET-CHAR     Se aduce caracterul prezent.
                CP    +29               Este o ')'?
                R     Z,2A48,SV-DIM     Salt înainte ca la terminarea
                                        contorizării elementelor.
                CP    +CC               Este 'TO'?
                JR    Z,29E0,SV-CH-ADD  Salt înapoi dacă se lucrează
                                        cu un 'segment'.
29FB SV-MULT    PUSH  BC                Se salvează contorul număr
                                        dimensiune si octetul
                                        discriminator.
                PUSH  HL                Salvare contor-element.
                CALL  2AEE,DE,(DE+1)    Se aduce o dimensiune mărime
                                        în DE.
                EX    (SP),HL           Se trece contorul în HL si
                                        se stochează indicatorul
                                        variabilei.
                EX    DE,HL             Contorul trece în DE si
                                        mărimea dimensiune în HL.
                CALL  2ACC,INT-EXP1     Se evaluează următoarea
                                        semnătură.
                JR    C,2A20,REPORT-3   Se dă eroare dacă iasă din
                                        interval.
                DEC   BC                Rezultatul evaluării este
                                        decrementat la fel ca si
                                        contorul ce contorizează
                                        elementele care intervin
                                        înaintea elementului
                                        specificat.
                CALL  2AF4,GET-HL*DE    Se multiplică contorul prin
                                        mărimea dimensiune.
                ADD   HL,BC             Se adună rezultatul lui
                                        'INT-EXP1' la prezentul
                                        contor.
                POP   DE                Se aduce indicatorul variabil
                POP   BC                Se aduce numărul dimensiune
                                        si octetul discriminator.
                DJNZ  29C3,SV-COMMA     Se ciclează bucla pînă cînd
                                        'B' devine egal cu zero.
```

Fanionul SYNTAX/RUN este verificat înainte de a se separa multimile de siruri de multimile de numere.

```
                BIT   7,C               Se dă eroare dacă se verifică
2A12 SV-RPT-C   JR    NZ,2A7A,SL-RPT-C  sintaxa în acest punct.
                PUSH  HL                Salare contor.
                BIT   6,C               Salt înainte dacă se tratează
                JR    NZ,2A2C,SV-ELEM$  o multime de siruri.
```

Cînd se lucrează cu cu o multime de numere caracterul prezent trebuie să fie o ')'.

```
                LD    B,D               Se transferă indicatorul
                LD    C,E               variabil în registrul pereche
                                        BC.
                RST   0018,GET-CHAR     Se aduce caracterul prezent.
                CP    +29               Este o ')'?
                JR    Z,2A22,SV-NUMBER  Salt peste prezentare eroare
                                        numai dacă este necesar.
```

Prezentarea 3- Semnătură afară din interval

```
2A20 REPORT-3   RST   0008,ERROR-1      Se apelează rutina de tratare
                DEFB  +02               eroare.
```

Adresa locatiei dinainte de actuala formă în punct flotant poate fi acum calculată.

```
2A22 SV-NUMBER  RST   0020,NEXT-CHAR    Avans CH-ADD.
                POP   HL                Se aduce contorul.
```

```
                LD    DE,+0005          Există cinci octeți pentru
                                        fiecare element dintr-o
                                        multime de numere.
                CALL  2AF4,GET-HL*DE    Se calculează numărul total
                                        al octetilor dinainte de
                                        elementul cerut.
                ADD   HL,BC             HL va indica locatia dinainte
                                        de elementul cerut.
                RET                     Revenire cu această adresă.
```

Cîn se lucrează cu o multime de siruri lungimea unui element este dată de ultima 'mărime-dimensiune'. Se calculează parametrii caracteristici si apoi se trec în stiva calculatorului.

```
2A2C SV-ELEM$   CALL  2AEE,DE,(DE+1)    Se aduce ultima mărime -
                                        dimensiune.
                EX    (SP),HL           Indicatorul variabil trece în
                                        stivă si contorul trece în
                                        HL.
                CALL  2AF4,GET-HL*DE    Se multiplică 'contorul' cu
                                        'marimea dimensiune'.
                POP   BC                Se aduce indicatorul variabil
                ADD   HL,BC             Aceasta face ca HL să indice
                                        locatia de dinainte de sir.
                INC   HL                Asa că se indică actualul
                                        'start'.
                LD    B,D               se transferă ultima mărime -
                LD    C,E               dimensiune în BC pentru a
                                        forma 'lungimea'.
                EX    DE,HL             Se mută 'startul' în DE.
                CALL  2AB1,STK-ST-0     Se trec acesti parametrii în
                                        stiva calculatorului. Notă:
                                        Primul parametru este zero
                                        cînd indică un sir dintr-o
                                        'multime de siruri' si deci
                                        intrarea existentă nu va
                                        trebui refăcută.
```

Există trei forme posibile pentru ultima semnătură. Prima este ilustrată de - A$(2,4 TO 8) -, a doua de - A$(2)(4 TO 8) - si a treia de - A$(2) - care este o formă absentă si indică faptul că întregul sir este cerut.

```
                RST   0018,GET-CHAR     Se aduce prezentul caracter.
                CP    +29               Este o ')'?
                JR    Z,2A48,SV-DIN     Salt dacă este.
                CP    +2C               Este o ','?
                JR    NZ,2A20,REPORT-3  Prezentare eroare dacă nu
                                        este.
2A45 SV-SLICE   CALL  2A52,SLICING      Se foloseste SLICING pentru a
                                        modifica setul de parametrii.
2A48,SV-DIN     RST   0020,NEXT-CHAR    Se aduce următorul caracter.
2A49'SV-SLICE?  CP    +28               Este o '('?
                JR    Z,2A45,SV-SLICE   Salt înapoi dacă trebuie
                                        considerat un 'segment'.
```

Se poate face o revenire cînd s-a terminat considerarea ultimei semnături.

```
                RES   6,(FLAGS)         Semnalizare - rezultat sir.
                RET                     Revenire cu parametrii
                                        sirului formînd o 'ultimă
                                        valoare' în stiva
                                        calculatorului.
```

THE 'SLICING' SUBROUTINE (SUBRUTINA 'SEGMENTARE')
Prezentul sir poate fi segmentat folosind această subrutină. Subrutina este introdusă cu parametrii sirului prezenti în vîrful stivei calculatorului si în registrii A, B, C, D & E. Initial se testează fanionul SYNTAX/RUN si parametrii sirului sînt adusi numai pentru o linie care a fost executată.

```
2A52 SLICING    CALL  2530,SYNTAX-Z       Se verifică fanionul.
                CALL  NZ,28FE,STK-FETCH   Se iau parametrii din stivă
                                          în 'timpul executiei'.
```

Se consideră posibilitatea'segmentului' ce a fost '()'.

```
                RST   0020,NEXT-CHAR      Se aduce următorul caracter.
                CP    +29                 Este o ')'?
                JR    Z,2AAD,SL-STORE     Salt înainte dacă este asa.
```

Inaintea executiei registrii sînt prelucrati dpà cum urmeazà:

```
                PUSH  DE                'Startul' trece în stiva masinii.
                XOR   A                 Registrul A este sters si
                PUSH  AF                salvat.
                PUSH  BC                Se salveazà 'lungimea' pentru
                                        scurt timp.
                LD    DE,+0001          Se presupune cà 'segmentul'
                                        urmeazà sà înceapà cu primul
                                        caracter.
                RST   0018,GET-CHAR     Se aduce primul caracter.
                POP   HL                se trece 'lungimea' în HL.
```

Acum se evalueazà primul parametru al 'segmentului'.

```
                CP    +CC               Este caracterul prezent 'TO'?
                JR    Z,2A81,SL-SECOND  Primul parametru, prin
                                        absentà, va fi '1' dacà se
                                        executà saltul.
                POP   AF                In acestà fazà A este zero.
                CALL  2ACD,INT-EXP2     BC va contine primul
                                        parametru. A va contine +FF
                                        dacà a fost o eroare 'afarà
                                        din interval'.
                PUSH  AF                Oricum se salevazà valoarea.
                LD    D,B               Se transferà primul parametru
                LD    E,C               în DE.
                PUSH  HL                Se salveazà 'lungimea' pentru
                                        scurt timp.
                RST   0018,GET-CHAR     Se aduce caracterul prezent.
                POP   HL                Se reface 'lungimea'.
                CP    +CC               Este caracterul prezent 'TO'?
                JR    Z,2A81,SL-SECOND  Dacà este asa, salt înainte
                                        pentru a lua în considerare a
                                        doilea parametru.
                CP    +29               altfel se aratà cà acolo este
2A7A SL-RPT-C   JP    NZ,1C8A,REPORT-C  o parantezà închisà.
```

In acest punct a fost identificat un 'segment' de un singur caracter, cum ar fi - A$(4).

```
                LD    H,D               Ultimul caracter al
                LD    L,E               segmentului este de asemenea
                                        primul caracter.
                JR    2A94,SL-DEFINE    Salt înainte.
```

Acum se evalueazà al doilea parametru al 'segmentului'.

```
2A81 SL-SECOND  PUSH  HL                Se salveazà 'lungimea' pentru
                                        scurt timp.
                RST   0020,NEXT-CHAR    Se aduce urmàtorul caracter.
                POP   HL                Se reface 'lungimea'.
                CP    +29               Este caracterul prezent ')'?
                JR    Z,2A94,SL-DEFINE  Salt dacà nu existà un al
                                        doilea parametru.
                POP   AF                Dacà primul parametru a fost
                                        în interval A va contine zero
                CALL  2ACD,INT-EXP2     BC va contine al doilea
                                        parametru.
                PUSH  AF                Salvare 'registru eroare'.
                RST   0018,GET-CHAR     Se aduce caracterul prezent.
                LD    H,B               Se trece rezultatul obtinut
                LD    L,C               din INT-EXP2 în registrul
                                        pereche HL.
                CP    +29               Se verificà dacà este o
                JR    NZ,2A7A,SL-RPT-C  parantezà închisà.
```

Acum se definesc 'noii' parametrii.

```
2A94 SL-DEFINE  POP   AF                Se aduce 'registrul erorii'.
                EX    (SP),HL           Al doilea parametru trece în
                                        stivà iar 'startul' trece în
                                        HL.
                ADD   HL,DE             Primul parametru este adunat
                                        cu 'startul'.
                DEC   HL                Intoarcere cu o locatie
                                        pentru corectare.
                EX    (SP),HL           'Noul start' trece în stivà
```

```
                                                    iar al doilea parametru trece
                                                    în HL.
                AND   A                             Se scade primul parametru din
                SBC   HL,DE                         al doilea parametru pentru a
                                                    afla 'lungimea' segementului.
                LD    BC,+0000                      Se initializează 'lungimea
                                                    nouă'.
                JR    C,2AA8,SL-OVER                Un 'segment' negativ
                                                    este un 'sir nul' mai degrabă
                                                    decît o conditie de eroare
                                                    (vezi manual).
                INC   HL                            Incuviintarea octetului
                                                    inclusiv.
                AND   A                             Doar acum se testează
                                                    'registrul eroare'.
                JP    M,2A20,REPORT-3               Salt dacă fiecare parametru a
                                                    format în afara intervalului
                                                    sirului.
                LD    B,H                           Se transferă 'noua lungime'
                LD    C,L                           în BC.
2AA8 SL-OVER    POP   DE                            Se aduce 'noul start'.
                RES   6,(FLAGS)                     Asigurare că încă se mai
                                                    indică un sir.
2AAD SL-STORE   CALL  2530,SYNTAX-Z                 Revenire în acest punct dacă
                RET   Z                             se verifică sintaxa; altfel
                                                    se continuă în subrutina STK-
                                                    STORE.
```

THE 'STK-STORE' SUBROUTINE

Această subrutină trece valorile continute în registrii A, B, C, D & E în stiva calculatorului. Astfel stiva creste ca mărime cu 5 octeti cu fiecare apelare a acestei subrutine.

In mod normal subrutina se foloseste pentru a transfera parametrii sirului dar ea se mai foloseste si se STACK-BC si LOG (2↑A) pentru a transfera 'numere întregi mici' în stivă.

De notat că atunci cînd se stochează parametrii unui sir, prima valoare stocată (venită din registrul A) va fi zero dacă sirul vine dintr-o multime de siruri sau este un 'segemnt' al unui sir. Valoarea pentru un sir simplu complet va fi '1'. Acest 'fanion' este folosit în rutina de comandă 'LET' cînd '1' semnalizează că vechea copie a sirului trebuie 'refăcută'.

```
2AB1 STK-ST-0   XOR   A                             Semnalizare - un sir dintr-o
                                                    multime de siruri sau un
                                                    'segment' dintr-un sir.
2AB2 STK-STO-$  RES   6,(FLAGS)                     Asigurare că fanionul indică
                                                    un rezultat sir.
2AB6 STK-STORE  PUSH  BC                            Se salvează pentru scurt timp
                                                    B & C.
                CALL  33A9,TEST-5-SP                Este loc pentru 5 octeti?
                POP   BC                            Se reface B & C.
                LD    HL,(STKEND)                   Se aduce adresa primei
                                                    locatii de deasupra stivei
                                                    prezente.
                LD    (HL),A                        Se transferă primul octet.
                INC   HL                            Se trece mai departe.
                LD    (HL),E                        Se transferă al doilea si al
                INC   HL                            treilea octet; pentru un sir
                LD    (HL),D                        acestia vor reprezenta
                                                    'startul'.
                INC   HL                            Se trece mai departe.
                LD    (HL),C                        Se transferă al patrulea si
                INC   HL                            al cincilea octet; pentru un
                LD    (HL),B                        sir acestia vor reprezenta
                                                    'lungimea'.
                INC   HL                            Se trece mai departe astfel
                                                    încît să se indice locatia de
                                                    deasupra stivei.
                LD    (STKEND),HL                   Se salvează această adresă în
                RET                                 STKEND si se revine.
```

THE 'INT-EXP' SUBROUTINE

Această subrutină revine cu rezultatul evaluării 'expresiei următoare' ca o valoare întreagă continută în registrul pereche HL. De asemenea subrutina compară acest rezultat cu o valoare limită înlocuită în registrul pereche HL. Fanionul de transport va fi resetat dacă este o eroare 'în afara intervalului'.

Registrul A este folosit ca un 'registru de eroare' si contine +00 dacă nu este 'eroare anterioară' si +FF dacă a fost una.

```
2ACC INT-EXP1     XOR   A                    Se sterge 'registrul de
                                             eroare'.
2ACD INT-EXP2     PUSH  DE                   Se salveazá pe tot parcursul
                  PUSH  HL                   registrii pereche DE & HL.
                  PUSH  AF                   Se salveazá temporar
                                             'registrul de eroare'.
                  CALL  1C82,EXPT-1NUM       Se evalueazá 'expresia
                                             urmátoare pentru a obtine în
          valoare'.                          stiva calculatoului 'ultima
                  POP   AF                   Se reface 'registrul de
                                             eroare'.
                  CALL  2530,SYNTAX-Z        Salt înainte dacá se verificá
                  JR    Z,2AE8,I-RESTORE     sintaxa.
                  PUSH  AF                   Se salveazá din nou registrul
                                             de eroare.
                  CALL  1E99,FIND-INT2       'Ultima valoare' este
                                             comprimatá în BC.
                  POP   DE                   Registrul de eroare în DE.
                  LD    A,B                  O 'expresie urmátoare' care
                  OR    C                    dá zero este întotdeauna o
                  SCF                        eroare, asa cá salt înainte
                  JR    Z,2AE8,I-CARRY       dacá este asa.
                  POP   HL                   Se face o copie a valorii
                  PUSH  HL                   limitá. Aceastá va fi o
                                             'márime dimensiune', o 'DIM
                                             limit' sau o 'lungime sir'.
                  AND   A                    Acum se comparà rezultatul
                  SBC   HL,BC                evaluárii expresiei cu limita
```

Acum se modificá starea fanionului de transport si valoarea continutá în registrul D astfel încît sá se obtinà vloarea corespunzátoare pentru 'registrul de eroare'.

```
2AE8 I-CARRY      LD    A,D                  Se aduce 'vechea valoare a
                                             erorii'.
                  SBC   A,+00                Se formeazá 'noua valoare a
                                             erorii'; +00 dacá nu este
                                             niciodatá eroare / +FF sau
                                             mai micá dacá s-a gásit o
                                             eroare 'în afara
                                             intervalului' în aceastá
                                             trecere sau într-una
                                             anterioará.
```

Inainte de revenire se refac registrii.

```
2AEB I-RESTORE    POP   HL                   Se refac HL & DE.
                  POP   DE
                  RET                        Revenire; 'registrul de
                                             eroare' este registrul A.
```

THE 'DE,(DE+1)' SUBROUTINE
Aceastá subrutiná executá structura - LD DE,(DE+1) - si revine cu HL indicînd 'DE+2'.

```
2AEE DE,(DE+1)    EX    DE,HL                Pentru construire se
                                             foloseste HL.
                  INC   HL                   Se indicá 'DE+1'.
                  LD    E,(HL)               De fapt - LD E,(DE+1).
                  INC   HL                   Se indicá 'DE+2'.
                  LD    D,(HL)               De fapt - LD D,(DE+2).
                  RET                        Sfîrsit.
```

THE 'GET-HL*DE' SUBROUTINE
Numai dacá sintaxa a fost verificatá aceastá subrutiná apeleazá 'HL=HL*DE' care executá structura implicit.
Excesul celor 16 bit accesibili în registrul pereche HL dá prezentarea 'afará din memorie'. Aceasta nu este adevárata situatie dar implicá faptul cá memoria nu este suficient de mare pentru taskul examinat de programator.

```
2AF4 GET-HL*DE    CALL  2530,SYNTAX-Z        Revenire directá dacá s-a
                  RET   Z                    verificat sintaxa.
                  CALL  30A9,HL=HL*DE        Se executá înmultirea.
                  JP    C,1F15,REPORT-4      Prezentare 'Afará din
                                             memorie'.
                  RET                        Sfîrsit.
```

THE 'LET' COMMAND ROUTINE (RUTINA DE COMANDA 'LET')
Aceasta este rutina actuală de desemnare a comenzilor LET, READ si INPUT.

Dacă variabila destinatie este o 'variabilă nou declarată', atunci DEST va indica prima literă a numelui variabilei asa cum intervine în linia BASIC. Bitul 1 al lui FLAGX va fi setat.

Oricum, dacă variabila destinatie 'deja există', atunci bitul 1 al lui FLAGX va fi resetat si DEST va indica o variabilă numerică în locatia dinaintea celor cinci octeti ai 'vechiului număr'; si pentru o variabilă sir va indica prima locatie a 'sirului vechi'. Utilizarea lui DEST în acest fel se aplică la variabile simple si la elementele unei multimi.

Bitul 0 al lui FLAGX este setat dacă variabila destinatie este o variabilă sir simplu 'complet'. (Semnalizare - se sterge copia veche.)

Initial, valoarea curentă a lui DEST este colectată si se testează bitul 1 al lui FLAGS.

```
2AFF LET        LD    HL,(DEST)          Se aduce adresa prezentă în
                                         DEST.
                BIT   1,(FLAGX)          Salt dacă se tratează o
                JR    Z,2B66,L-EXISTS    variabilă care 'există deja'.
```

Se foloseste o 'variabilă nou declarată'. Asa că mai întîi se găseste lungimea numelui său.

```
                LD    BC,+0005           Se presupune că se lucrează
                                         cu o varibilă numerică -
                                         5 octeti.
```

Se introduce o buclă care va lucra cu caracterele unui nume lung. Se ignoră orice cod de culoare sau spatiu din nume.

```
2B0B L-EACH-CH  INC   BC                 Se adună '1' la contor pentru
                                         fiecare caracter al numelui.
2B0C L-NO-SP    INC   HL                 Se trece de-a lungul numelui
                                         variabilei.
                LD    A,(HL)             Se aduce 'codul prezent'.
                CP    +20                Salt înapoi dacă este un
                JR    Z,2B0C,L-NO-SP     'spatiu'; astfel se ignoră
                                         spatiile.
                JR    NC,2B1F,L-TEST-CH  Salt înainte dacă codul este
                                         +21 la +FF.
                CP    +10                Se acceptă, ca si cod final,
                JR    C,2B29,L-SPACES    cel din intervalul +00 la +FF
                CP    +16                De asemenea se acceptă si
                JR    NC,2B29,L-SPACES   intervalul +16 la +1F.
                INC   HL                 Se trece peste codul de
                                         control după fiecare INK la
                                         OVER.
                JR    2B0C,L-NO-SP       Salt înapoi ca si cum aceste
                                         coduri de control sînt
                                         tratate ca spatii.
```

Se separă numele 'numeric' si 'sir'.

```
2B1F L-TEST-CH  CALL  2C88,ALPHANUM      Este cod alfanumeric?
                JR    C,2B0B,L-EACH-CH   Dacă da, atunci se acceptă ca
                                         si cum ar fi un caracter al
                                         unui nume 'lung'.
                CP    +24                Este codul prezent '$'?
                JP    Z,2BC0,L-NEW$      Salt înainte ca si cum s-ar
                                         trata un sir simplu 'nou
                                         declarat'.
```

'Variabila numerică nou declarată' tratată în prezent va cere 'BC' spatii în spatiul variabilelor pentru numele si valoarea sa. Spatiul este accesat si numele variabilei este copiat peste cu caracterele 'marcate' cum s-a cerut.

```
2B29 L-SPACES   LD    A,C                Se copiază 'lungimea' în A.
                LD    HL,(E-LINE)        HL va indica al 80-lea octet
                DEC   HL                 la sfîrsitul zonei
                                         variabilelor.
                CALL  1655,MAKE-ROOM     Acum se deschide zona
                                         variabilelor.
                                         Notă: De fapt spatiile 'BC'
                                         se fac înainte de a înlocui
                                         'octetul 80'.
                INC   HL                 Se indică primul octet 'nou',
                INC   HL                 DE va indica al doilea octet
                EX    DE,HL              'nou'.
                PUSH  DE                 Se salvează acest indicator.
                LD    HL,(DEST)          Se aduce indicatorul la
                                         începutul numelui.
```

```
                DEC   DE                DE va indica primul octet
                                        'nou'.
                SUB   +06               B va contine 'numărul de
                LD    B,A               litere aditionale' care sînt
                                        găsite într-un 'nume lung'.
                JR    Z,2B4F,L-SINGLE   Salt înainte dacă se lucrează
                                        cu o variabilă cu 'nume
                                        scurt'.
```

Codurile 'aditionale' ale unui nume lung sînt trecute în spatiul variabilelor.

```
2B3E L-CHAR     INC   HL                Se indică fiecare cod
                                        'aditional'.
                LD    A,(HL)            Se aduce codul.
                CP    +21               Se acceptă coduri de la +21
                JR    C,2B3E,L-CHAR     la +FF; se ignoră codurile
                                        de la +00 la +20.
                OR    +20               Se setează bitul 5 pentru
                                        portiunea cu litere mici.
                INC   DE                Se transferă pe rînd codurile
                LD    (DE),A            în al 2-lea octet 'nou' în
                                        comtinuare.
                DJNZ  2B3E,L-CHAR       Se ciclează bucla pentru
                                        toate codurile 'aditionale'.
```

Ultimul cod al unui nume 'lung' trebuie să fie făcut SAU (OR) cu +80.

```
                OR    +80               Se marchează codul cum se
                LD    (DE),A            cere si se supraimprimă
                                        ultimul cod.
```

Se consideră acum prima literă a numelui variabilei ce a fost tratată.

```
                LD    A,+C0             Pregătire pentru marcarea
                                        literei unui 'nume' lung.
284F L-SINGLE   LD    HL,(DEST)         Se aduce indicatorul la
                                        literă.
                XOR   (HL)              A contine +00 pentru un nume
                                        'scurt' si +C0 pentru un nume
                                        'lung'.
                OR    +20               Setare bit 5 pentru portiunea
                                        cu litere mici.
                POP   HL                Acum se coboară indicatorul.
```

Acum se apelează subrutina L-FIRST pentru a introduce 'litera' în locatia ei corespunzătoare.

```
                CALL  2BEA,L-FIRST      Se introduce litera si se
                                        revine cu HL indicînd 'noul
                                        octet 80'.
```

Acum se poate transfera 'ultima valoare' în spatiul variabilelor. De notat că în acest punct HL indică întotdeauna locatia de după cele cinci locatii alocate numărului.
Se foloseste o instructiune 'RST 0028' pentru a apela CALCULATOR si 'ultima valoare' este stearsă. Oricum această valoare nu este supraimprimată.

```
2B59 L-NUMERIC  PUSH  HL                Se salvează indicatorul
                                        'destinatie'.
                RST   0028,FP-CALC      Utilizare calculator.
                DEFB  +02,delete        Acesta mută STKEND înapoi cu
                DEFB  +38,end-calc      cinci octeti.
                POP   HL                Se reface indicatorul.
                LD    BC,+0005          Se dă numărului o 'lungime'
                                        de cinci octeti.
                AND   A                 HL va indica prima din cele
                SBC   HL,BC             cinci locatii si se execută
                JR    2BA6,L-ENTER      salt pentru a realiza
                                        actualul transfer.
```

Se vine aici dacă se consideră o variabilă care 'există deja'. Mai întîi se testează bitul 6 al lui FLAGS astfel încît să se separe variabilele numerice de variabile sir sau multimi de siruri.

```
2B66 L-EXISTS   BIT   6,(FLAGS)         Salt mai departe dacă se
                JR    Z,2B72,L-DELETE-$ tratează orice fel de
                                        variabilă sir.
```

Pentru variabile numerice numărul 'nou' se supraimprimă peste numărul

'vechi'. Asa cà mai întîi HL va trebuie sà indice locatia de dupà cei cinci octeti ai intrarii existente. In prezent HL indicà locatia de dinainte de cei cinci biti.

```
                LD    DE,+0006          Cei cinci octeti ai unui
                                        numàr + '1'.
                ADD   HL,DE             HL indicà acum 'dupà'.
                JR    2B59,L-NUMERIC    Salt înapoi pentru a efectua
                                        transferul actual.
```

Se aduc parametrii variabilei sir si sirurile simple complete se separà de sirurile 'segmentate' si multimi de siruri.

```
2B72 L-DELETE-$ LD    HL,(DEST)         Se aduce 'începutul'. Notà:
                                        Aceastà linie este de prisos.
                LD    BC,(STRLEN)       Se aduce 'lungimea'.
                BIT   0,(FLAGX)         Salt dacà se lucrează cu un
                JR    NZ,2BAF,L-ADD$    sir simplu complet; doar în
                                        acest caz este necesar sà se
                                        steargà sirul 'vechi'.
```

Cînd se lucrează cu un segment al unui sir simplu existent, un 'segment' al sirului dintr-o multime de siruri sau un sir complet dintr-o multime de siruri existà douà faze distincte. In prima fazà se construieste 'noul' sir în spatiul de lucru, lungindu-l sau scurtîndu-l, dupà cum se cere. In a doua fazà se copiazà 'noul' sir în spatiul alocat lui în spatiul variabilelor.

```
                LD    A,B               Revenire dacà sirul este
                OR    C                 invalidat.
                RET   Z
```

Apoi se realizeazà numàrul cerut de spatii accesibile în spatiul de lucru.

```
                PUSH  HL                Salvare 'început' (DEST).
                RST   0030,BC-SPACES    Se realizează spatiul total
                                        necesar în spatiul de lucru.
                PUSH  DE                Se salvează indicatorul
                                        primei locatii.
                PUSH  BC                Se salvează 'lungiea' pentru
                                        a fi folosità mai tîrziu.
                LD    D,H               DE va indica ultima locatie.
                LD    E,L
                INC   HL                HL va indica 'una peste'
                                        locatia nouà.
                LD    (HL),+20          Se introduce un caracter
                                        'spatiu'
                LDDR                    Se copiazà acest caracter în
                                        toate locatiile noi. Se
                                        terminà cu HL indicînd prima
                                        locatie nouà.
```

Acum se aduc din stiva calculatorului parametrii sirului care a fost tratat.

```
                PUSH  HL                Se salvează indicatorul
                                        pentru scurt timp.
                CALL  2BF1,STK-FETCH    Se aduc parametrii 'noi'.
                POP   HL                Se reface indicatorul
```

Notà: In acest punct necesarul total de spatii a fost fàcut accesibil în spatiul de lucru pentru 'variabilele în atribuire'. De exemplu - pentru instructiunea - LET A$(4 TO 8)="abcdefg" - au fost fàcute cinci locatii.

Parametrii încàrcati deasupra ca o 'ultimà valoare' reprezintà sirul care trebuie copiat în noile locatii cu fortarea lungirii sau scurtàrii, dupà cum se cere.

Lungimea sirului 'nou' se comparà cu lungimea spatiului fàcut accesibil pentru el.

```
                EX    (SP),HL           'Lungimea' noii zone în HL.
                                        'Indicatorul' noii zone în
                                        stivà.
                AND   A                 Se compară cele douà
                SBC   HL,BC             'lungimi' si se executà salt
                ADD   HL,BC             înainte dacà 'noul' sir va
                                        corespunde spatiului.
                JR    NC,2B9B,L-LENGTH  Aceasta înseamnà cà nu se
                                        cere scurtarea.
                LD    B,H               Oricum se modificà 'noua'
                LD    C,L               lungime dacà este prea mare.
2B9B L-LENGTH   EX    (SP),HL           'Lungimea' noii zone în
```

```
                                                    stivă.
                                                    Indicatorul' noii zone în HL.
```

Atîta timp cît noul sir nu este un 'sir invalidat' el este copiat în spatiul de lucru. Lungirea fortată se realizează automat dacă 'noul' sir este scurt decît spatiul accesat pentru acesta.

```
                EX    DE,HL                 'Inceputul' sirului nou în HL
                                            'Indicatorul' noii zone în DE
                LD    A,B                   Salt înainte dacă sirul 'nou'
                OR    C                     este un sir invalidat.
                JR    Z,2BA3,L-IN-W/S
                LDIR                        Altfel se mută sirul 'nou' în
                                            spatiul de lucru.
```

Se refac valorile care au fost salvate în stiva masinii.

```
2BA3 L-IN-W/S   POP   BC                    'Lungimea' zonei noi.
                POP   DE                    'Indicatorul' zonei noi.
                POP   HL                    'Inceputul' - indicatorul
                                            'variabilei în atribuire'
                                            care a fost initial în DEST.
                                            Acum se foloseste L-ENTER
                                            pentru a trece sirul 'nou' în
                                            spatiul variabilelor.
```

THE 'L-ENTER' SUBROUTINE

Această scurtă subrutină este folosită pentru a trece fie o valoare numerică, din stiva calculatorului, sau un sir, din spatiul de lucru, în pozitia sa corespunzătoare în spatiul variabilelor.

De aceea această subrutină este folosită pentru toate valorile numerice si sirurile, exceptînd sirurile simple 'nou declarate' si sirurile simple 'complete & existente'.

```
2BA6 L-ENTER    EX    DE,HL                 Schimbare între indicatori.
                LD    A,B                   Se verifică încă o dată că
                OR    C                     lungimea este diferită de
                RET   Z                     zero.
                PUSH  DE                    Salvare indicator destinatie.
                LDIR                        Se transferă valoarea
                                            numerică sau sirul.
                POP   HL                    Revenire cu registrul pereche
                RET                         HL indicînd primul octet al
                                            valorii numerice sau al
                                            sirului.
```

THE LET SUBROUTINE CONTIUNUES HERE (SUBRUTINA LET CONTINUA AICI)

Cînd se tratatează un sir simplu 'complet & existent' noul sir este introdus ca si cum ar fi fost un sir simplu 'nou declarat' înainte ca versiunea existentă să fie cerută.

```
2BAF L-ADD$     DEC   HL                    HL va indica litera numelui
                DEC   HL                    variabilei.
                DEC   HL                    Aceasta înseamnă DEST - 3.
                LD    A,(HL)                Se extrage litera.
                PUSH  HL                    Se salvează indicatorul
                                            'versiunii existente'.
                PUSH  BC                    Se salvează 'lungimea'
                                            'sirului existent'.
                CALL  2BC6,L-STRING         Se foloseste L-STRING pentru
                                            a adăuga noul sir spatiului
                                            variabilelor.
                POP   BC                    Se readuce 'lungimea'.
                POP   HL                    Se readuce indicatorul.
                INC   BC                    Este permis un octet pentru
                INC   BC                    literă si doi octeti pentru
                INC   BC                    lungime.
                JP    19E8,RECLAIM-2        Iesire prin salt la RECLAIM-2
                                            care va cere întreaga
                                            versiune existentă.
```

Sirurile simple 'nou declarate' sînt tratate după cum urmează:

```
2BC0 L-NEW$     LD    A,+DF                 Pregătire marcare literă
                                            variabilă.
                LD    HL,(DEST)             Se aduce indicatorul pe
                                            literă.
                AND   (HL)                  Se marchează litera asa cum
                                            se cere. L-STRING se
```

```
                                                  foloseste acum pentru a
                                                  adauga noul sir spatiului
                                                  variabilelor.
```

THE 'L-STRING' SUBROUTINE
Se aduc parametrii 'noului' sir, se face accesibil suficient spatiu pentru el si apoi se transferă sirul.

```
2BC6 L-STRING     PUSH  AF                  Se salvează litera variabilei
                  CALL  2BF1,STK-FETCH      Se aduce 'începutul' si
                                            'lungimea' sirului 'nou'.
                  EX    DE,HL               Se mută 'începutul' în HL.
                  ADD   HL,BC               HL va indica 'unu peste' sir.
                  PUSH  BC                  Salvare 'lungime'.
                  DEC   HL                  HL va indica sfîrsitul
                                            sirului.
                  LD    (DEST),HL           Se salvează indicatorul
                                            pentru scurt timp.
                  INC   BC                  Este permis un octet pentru
                  INC   BC                  literă si doi octeti pentru
                  INC   BC                  lungime.
                  LD    HL,(E-LINE)         HL va indica al'80-lea octet'
                  DEC   HL                  la sfîrsitul zonei
                                            variabilelor.
                  CALL  1655,MAKE-ROOM      Acum se deschide zona
                                            variabilelor.
                                            Notă: De fapt 'BC' spatii
                                            sînt făcute înaintea
                                            înlocuirii 'octetului 80'.
                  LD    HL,(DEST)           Se readuce indicatorul la
                                            sfîrsitul sirului 'nou.
                  POP   BC                  Se face o copie a lungimii
                  PUSH  BC                  sirului 'nou'.
                  INC   BC                  Se adună unu la lungime dacă
                                            sirul 'nou' este un sir
                                            'nul'.
                  LDDR                      Acum se copiază 'noul' sir +
                                            un octet.
                  EX    DE,HL               HL va indica octetul care
                  INC   HL                  contine partea cea mai
                                            semnificativă a lungimii.
                  POP   BC                  Se aduce 'lungimea'.
                  LD    (HL),B              Se introduce partea cea mai
                                            semnificativă a lungimii.
                  DEC   HL                  Inapoi cu unu.
                  LD    (HL),C              Se introduce partea cea mai
                                            putin semnificativă a
                                            lungimii.
                  POP   AF                  Se aduce litera variabilei.
```

THE 'L-FIRST' SUBROUTINE
Această subrutină este introdusă cu litera variabilei, convenabil marcată, în registrul A. Litera este suprapusă peste al '80-lea octet' în spatiul variabilelor. Subrutina revine cu registrul pereche HL indicînd 'noul octet 80'.

```
2BEA L-FIRST      DEC   HL                  HL va indica 'vechiul octet
                                            80'.
                  LD    (HL),A              Acesta este suprapus cu
                                            litera variabilei.
                  LD    HL,(E-LINE)         HL va indica 'noul octet 80'.
                  DEC   HL
                  RET                       Se termină cu toate
                                            'variabilele nou declarate'.
```

THE 'STK-FETCH' SUBROUTINE
Acesta subrutină importantă colectează 'ultima valoare' din stiva calculatorului. Cei cinci octeti pot fi sau un număr în virgulă mobilă, în forma 'lungă' sau 'scurtă', sau un set de parametrii care definesc un sir.

```
2BF1 STK-FETCH    LD    HL,(STKEND)         Se aduce STKEND.
                  DEC   HL                  Inapoi cu unu.
                  LD    B,(HL)              A cincea valoarea.
                  DEC   HL                  Inapoi cu unu.
                  LD    C,(HL)              A patra valoarea.
                  DEC   HL                  Inapoi cu unu.
                  LD    D,(HL)              A treia valoare.
                  DEC   HL                  Inapoi cu unu.
                  LD    E,(HL)              A doua valoare.
```

```
                DEC   HL                    Inapoi cu unu.
                LD    A,(HL)                Prima valoare.
                LD    (STKEND),HL           Se resetează STKEND pe noua
                                            sa pozitie.
                RET                         Sfîrsit.
```

THE 'DIM' COMMAND ROUTINE (RUTINA DE COMANDA 'DIM')
Această rutină stabileste noi multimi în spatiul variabilelor. Rutina porneste căutînd spatiul variabilelor existent pentru a determina dacă este vreo multime existentă cu acelasi nume. Dacă s-a găsit o asemenea multime aceasta este refăcută înainte de stabilirea noii dispuneri.
O multime nouă va avea toate elementele setate pe zero, dacă este o multime numerică, sau pe 'spatii', dacă este o multime de siruri.

```
2C02 DIM        CALL  28B2,LOOK-VARS        Se caută în spatiul
                                            variabilelor.
2C05 D-RPORT-C  JP    NZ,1C8A,REPORT-C      Se dă prezentarea C ca si cum
                                            ar fi fost o eroare.
                CALL  2530,SYNTAX-Z         Salt înainte dacă este în
                JR    NZ,2C15,D-RUN         'timpul executiei'.
                RES   6,C                   Se verifică sintaxa pentru
                                            multimile sir ca si pentru
                                            cele numerice.
                CALL  2996,STK-VAR          Se verifică sintaxa
                                            expresiilor între paranteze.
                CALL  1BEE,CHECK-END        Se trece la considerarea
                                            următoarei instructiuni dacă
                                            sintaxa a fost
                                            satisfăcătoare.
```

Se cere o 'multime existentă'.

```
2C15 D-RUN      JR    C,2C1F,D-LETTER       Salt înainte dacă nu există
                                            nici o 'multime existentă'.
                PUSH  BC                    Se salvează octetul
                                            discriminator.
                CALL  19B8,NEXT-ONE         Se determină începutul
                                            următoarei variabile.
                CALL  19E8,RECLAIM-2        Se reface 'dispunerea
                                            existentă'.
                POP   BC                    Se reface octetul
                                            discriminator.
```

S-au găsit paramterii initiali ai noii multimi.

```
2C1F D-LETTER   SET   7,C                   Se setează bitul 7 al
                                            octetului discriminator.
                LD    B,+00                 Se pune contorul dimensiunii
                                            pe zero.
                PUSH  BC                    Se salvează contorul si
                                            octetul discriminator.
                LD    HL,+0001              Registrul pereche HL va
                BIT   6,C                   contine mărimea elementelor
                JR    NZ,2C2D,D-SIZE        din multime, '1' pentru o
                LD    L,+05                 multime de siruri/ '5' pentru
                                            o multime numerică.
2C2D D-SIZE     EX    DE,HL                 Mărime element în DE.
```

Bucla următoare este accesată pentru fiecare dimensiune care este specificată în expresia dintre paranteze a instructiunii DIM. Numărul total de octeti cerut pentru elementele multimii este construit în registrul pereche DE.

```
2C2E D-NO-LOOP  RST   0020,NEXT-CHAR       Avans CH-ADD la fiecare
                                            trecere.
                LD    H,+FF                 Se setează o 'valoare-limită'
                CALL  2ACC,INT-EXP1         Se evaluează un parametru.
                JP    C,2A20,REPORT-3       Se dă eroare dacă este 'în
                                            afara intervalului'.
                POP   HL                    Se aduce contorul-dimensiune
                                            si octetul discriminator.
                PUSH  BC                    Se salvează parametrul la
                                            fiecare trecere prin buclă.
                INC   H                     Se incrementează contorul -
                                            dimensiune la fiecare trecere
                                            prin buclă.
                PUSH  HL                    Se stochează din nou contorul
                                            dimensiune si octetul
                                            discriminator.
                LD    H,B                   Parametrul este mutat în
```

```
LD    L,C          registrul pereche HL.
CALL  2AF4,GET-HL*DE  Octetul total este construit
EX    DE,HL        în HL si apoi este transferat
                   în DE.
RST   0018,GET-CHAR  Se aduce prezentul caracter
CP    +2C          si se ciclează din nou bucla
JR    Z,2C2E,D-NO-LOOP  dacă mai există o altă
                   dimensiune.
```

Notă: In acest punct registrul pereche DE indică numărul de octeti cerut pentru elementele noii multimi si mărimea fiecărei dimensiuni este stocată în stiva masinii.

Acum se verifică dacă este într-adevăr o virgulă care închide la expresia dintre paranteze.

```
CP    +29          Este o ')'?
JR    NZ,2C05,D-RPORT-C  Salt înapoi dacă nu este.
RST   0020,NEXT-CHAR  Avans CH-ADD peste aceasta.
```

Acum este permisă aducerea mărimilor dimensiune.

```
POP   BC           Se aduce contorul dimensiune
                   si octetul discriminator.
LD    A,C          Se trece octetul în registrul
                   A octetul discriminator
                   pentru mai târziu.
LD    L,B          Se mută contorul în L.
LD    H,+00        Se sterge registrul H.
INC   HL           Se incrementează contorul -
INC   HL           dimensiune cu doi si se
ADD   HL,HL        dublează rezultatul si se
ADD   HL,DE        formează lungimea totală
                   corectă prin adăugarea
                   elementului octet total.
JP    C,1F15,REPORT-4  Se dă prezentarea 'Afară din
                   memorie' dacă se cere.
PUSH  DE           Salvare element octet total.
PUSH  BC           Se salvează contorul -
                   dimensiune si octetul
                   discriminator.
PUSH  HL           Se salvează lungimea totală.
LD    B,H          Se mută lungimea totală în BC
LD    C,L
```

Spatiul total cerut este făcut accesibil pentru noua multime la sfârsitul spatiului variabilelor.

```
LD    HL,(E-LINE)  Registrul pereche HL va
DEC   HL           indica 'bitul 80'.
CALL  1655,MAKE-ROOM  Spatiul este făcut accesibil.
INC   HL           HL va indica prima locatie
                   nouă.
```

Acum se introduc parametrii.

```
LD    (HL),A       Litera, convenabil marcată,
                   se introduce prima.
POP   BC           Se aduce lungimea totală si
DEC   BC           se micsorează cu '3'.
DEC   BC
DEC   BC
INC   HL           Avans HL.
LD    (HL),C       Se introduce partea cea mai
                   putin semnificativă a
                   lungimii.
INC   HL           Avans HL.
LD    (HL),B       Se introduce partea cea mai
                   semnificativă a lungimii.
POP   BC           Se aduce contorul-dimensiune.
LD    A,B          Acesta se pune în registrul A
INC   HL           Avans HL.
LD    (HL),A       Se introduce contorul -
                   dimensiune.
```

Acum se 'sterg' elementele noii multimi.

```
LD    H,D          HL va indica ultima locatie a
LD    L,E          multimii iar DE locatia de
DEC   DE           dinainte de aceasta.
```

```
                LD    (HL),+00          Se introduce un zero în
                BIT   6,C               ultima locatie dar se
                JR    Z,2C7C,DIM-CLEAR  suprapune un 'spatiu' dacă se
                LD    (HL),+20          lucrează cu o multime de
                                        siruri.
2C7C DIM-CLEAR  POP   BC                Se aduce întregul octet al
                                        elementului.
                LDDR                    Se sterge multimea + o
                                        locatie aditională.
```

Acum se introduc 'mărimile dimensiune'.

```
2C7F DIM-SIZES  POP   BC                Se aduce o mărime dimensiune.
                LD    (HL),B            Se introduce octetul cel mai
                                        semnificativ.
                DEC   HL                Inapoi cu unu.
                LD    (HL),C            Se introduce octetul mai
                                        putin semnificativ.
                DEC   HL                Inapoi cu unu.
                DEC   A                 Se decrementează contorul
                                        dimensiune.
                JR    NZ,2C7F,DIM-SIZES Se repetă operatia pînă cînd
                RET                     s-au luat în considerare
                                        toate dimensiunile; apoi se
                                        revine.
```

THE 'ALPHANUM' SUBROUTINE
Această subrutină revine cu fanionul de transport setat dacă valoarea prezentă a registrului A denotă o cifră sau o literă validă.

```
2C88 ALPHANUM   CALL  2D1B,NUMERIC      Testare cifră; trasportul va
                                        fi resetat dacă este o cifră.
                CCF                     Se complementează fanionul de
                                        transport.
                RET   C                 revenire dacă este cifră;
                                        altfel se continuă cu ALPHA.
```

THE 'ALPHA' SUBROUTINE
Această subrutină revine cu fanionul de transport setat dacă valoarea prezentă în registrul A denotă o literă validă din alfabet.

```
2C8D ALPHA      CP    +41               Se compară cu 41 hex. codul
                                        lui 'A'.
                CCF                     Se complementează fanionul de
                                        transport.
                RET   NC                Revenire dacă nu este un cod
                                        caracter valid.
                CP    +5B               Se compară cu 5B hex., 1 mai
                                        mult decît codul pentru 'Z'.
                RET   C                 Revenire dacă este majusculă.
                CP    +61               Se compară cu 61 hex., codul
                                        pentru 'a'.
                CCF                     Se complementează fanionul de
                                        transport.
                RET   NC                Revenire dacă nu este codul
                                        unui caracter valid.
                CP    +7B               Se compară cu 7B hex., cu 1
                                        mai mult decît codul pentru
                                        'z'.
                RET                     Sfîrsit.
```

THE 'DECIMAL TO FLOATING POINT' SUBROUTINE (SUBRUTINA 'ZECIMAL IN VIGULA MOBILA')
Ca parte a verificării sintaxei, numerele zecimale care intervin într-o linie BASIC sînt convertite în forma lor în virgulă mobilă. Această subrutină citeste numărul zecimal cifră cu cifră si dă rezultatul ca o 'ultimă valoare' în stiva calculatorului. Dar mai întîi ea lucrează cu notatia alternativă BIN, care introduce o secevntă de 0 si 1 dînd reprezentarea binară a numărului cerut.

```
2C9B DEC-TO-FP  CP    +C4               Este caracterul un 'BIN'?
                JR    NZ,2CB8,NOT-BIN   Salt dacă nu este 'BIN'.
                LD    DE,+0000          Initializare rezultat pe zero
                                        în DE.
2CA2 BIN-DIGIT  RST   0020,NEXT-CHAR    Se aduce următorul caracter.
                SUB   +31               Se scade '1' din codul
                                        caracterului.
                ADC   A,+00             0 dă acum 0 cu trasportul
                                        setat; 1 dă 0 cu transportul
```

```
                        resetat.
        JR    NZ,2CB3,BIN-END   Orice alt caracter cauzează
                                salt la BIN-END si se va
                                verifica sintaxa în timpul
                                sau după baleiere.
        EX    DE,HL             Deocamdată rezultatul este în
                                HL.
        CCF                     Se complementează fanionul de
                                transport.
        ADC   HL,HL             Se deplasează rezultatul la
                                stînga, cu transportul
                                trecînd în bitul 0.
        JP    C,3IAD,REPORT-6   Se prezintă depăsire dacă
                                este mai mare decît 65535.
        EX    DE,HL             Se readuce deocamdată
                                rezultatul în DE.
        JR    2CA2,BIN-DIGIT    Salt înapoi pentru următorul
                                0 sau 1.
2CB3 BIN-END
        LD    B,D               Se copiază rezultatul în BC
        LD    C,E               pentru a fi stocat.
        JP    2D2B,STACK-BC     Salt înainte pentru a stoca
                                rezultatul.
```

Pentru celelalte numere, mai întîi se converteste partea întreagă; dacă următorul caracter este unul zecimal, atunci se consideră fractie zecimală.

```
2CB8 NOT-BIN
        CP    +2E               Este primul caracter un '.'?
        JR    Z,2CCB,DECIMAL    Salt mai departe dacă este.
        CALL  2D3B,INT-TO-FP    Altfel, se formează o 'ultimă
                                valoare' a întregului.
        CP    +2E               Este următorul caracter un
                                '.'?
        JR    NZ,2CEB,E-FORMAT  Salt înainte pentru a vedea
                                dacă este un 'E'.
        RST   0020,NEXT-CHAR    Se aduce următorul caracter.
        CALL  2D1B,NUMERIC      Este o cifră?
        JR    C,2CEB,E-FORMAT   Salt dacă nu (De exemplu este
                                permis 1.E4).
        JR    2CD5,DEC-STO-1    Salt înainte pentru a
                                prelucra cifrele de după
                                virgulă.
2CCB DECIMAL
        RST   0020,NEXT-CHAR    Dacă numărul începe cu un
        CALL  2D1B,NUMERIC      zecimal, se verifică dacă
                                următorul caracter este o
                                cifră.
2CCF DEC-RPT-C
        JP    C,1C8A,REPORT-C   Dacă nu este, se dă eroare.
        RST   0028,FP-CLAC      Se utilizează calculatorul
        DEFB  +A0,stk-zero      pentru a stoca zero ca parte
        DEFB  +38,end-calc      întreagă a unui astfel de
                                număr.
2CD5 DEC-STO-1
        RST   0028,FP-CALC      Utilizare calculator.
        DEFB  +A1,stk-one       Se caută forma în virgulă
        DEFB  +C0,st-mem-0      mobilă a numărului zecimal
        DEFB  +02,delete        '1' si se salvează în spatiul
        DEFB  +38,end-calc      memoriei.
2CDA NXT-DGT-1
        RST   0018,GET-CHAR     Se aduce caracterul prezent.
        CALL  2D22,STK-DIGIT    Acesta se stochează dacă este
                                cifră.
        JR    C,2CEB,E-FORMAT   Dacă nu este, salt înainte.
        RST   0028,FP-CALC      Utilizare calculator.
        DEFB  +E0,get-mem-0     Pentru fiecare trecere prin
        DEFB  +A4,stk-ten       buclă, numărul salvat în
        DEFB  +05,division      spatiul memoriei este adus,
        DEFB  +C0,st-mem-0      împărtit cu 10 si restocat;
                                aceasta înseamnă plecînd de
                                la .1 la .01 la.001 etc.
        DEFB  +04,multipy       Cifra prezentă este înmultită
        DEFB  +0F,addition      cu 'numărul salvat' si
        DEFB  +38,end-calc      adunată cu 'ultima valoare'.
        RST   0020,NEXT-CHAR    Se aduce următorul caracter.
        JR    2CDA,NXT-DGT-1    Salt înapoi pentru a-l lua în
                                considerare (cu un octet mai
                                mult decît este necesar).
```

In continuare se consideră orice 'notatie E', de exemplu forma xEm sau xem, unde m un întreg pozitiv sau negativ.

```
2CEB E-FORMAT
        CP    +45               Este caracterul prezent un
                                'E'?
```

```
                JR    Z,2CF2,SIGN-FLAG      Salt înainte dacă este.
                CP    +65                   Este un 'e'?
                RET   NZ                    Da, terminare.
2CF2 SIGN-FLAG  LD    B,+FF                 Se foloseste B ca un fanion
                                            de semn, FF pentru +.
                RST   0020,NEXT-CHAR        Se aduce următorul caracter.
                CP    +2B                   Este un '+'?
                JR    Z,2CFE,SIGN-DONE      Salt înainte.
                CP    +2D                   Este '-'?
                JR    NZ,2CFF,ST-E-PAT      Salt dacă nu este nici '+'
                                            nici '-'.
                INC   B                     Se schimbă semnul fanionului.
2CFE SIGN-DONE  RST   0020,NEXT-CHAR        Se indică prima cifră.
2CFF ST-E-PART  CALL  2D1B,NUMERIC          Este într-adevăr o cifră?
                JR    C,2CCF,DEC-RPT-C      Se prezintă eroare dacă nu
                                            este.
                PUSH  BC                    Se salvează fanionul în B
                                            pentru scurt timp.
                CALL  2D3B,INT-TO-FP        Se stochează ABS m, unde m
                                            este exponentul.
                CALL  2DD5,FP-TO-A          Se transferă ABS m în A.
                POP   BC                    Se reface fanionul de semn în
                                            B.
                JP    C,31AD,REPORT-6       Acum se prezintă depăsire
                AND   A                     dacă ABS m este mai mare
                JP    M,31AD,REPORT-6       decît 255 sau într-adevăr mai
                                            mare decît 127 (celelalte
                                            valori mai mari decît 39 vor
                                            fi detectate ai tîrziu).
                INC   B                     Se testează fanionul de semn
                                            în B: '+' (de fapt +FF) va
                                            seta acum fanionul de zero.
                JR    Z,2D18,E-FP-JUMP      Salt dacă semnul lui m este
                                            '+'.
                NEG                         Se face negarea lui m dacă
                                            semnul lui este '-'.
2D18 E-FP-JUMP  JP    2D4F,E-TO-FP          Salt pentru a atribui
                                            'ultimei valori' rezultatul
                                            lui x*10 ^ m.
```

THE 'NUMERIC' SUBROUTINE (SUBRUTINA 'NUMERIC')
Această subrutină revine cu fanionul de transport resetat dacă valoarea prezentă în registrul A denotă o cifră validă.

```
2D1B NUMERIC    CP    +30                   Se compară cu 30 hex, codul
                                            pentru '0'.
                RET   C                     Revenire dacă nu este codul
                                            unui caracter valid.
                CP    +3A                   Se compară cu limita
                                            superioară.
                CCF                         Se complementează fanionul de
                                            transport.
                RET                         Sfîrsit.
```

THE 'STK-DIGIT' SUBROUTINE (SUBRUTINA 'STK-DIGIT')
Această subrutină revine imediat dacă valoarea curentă continută în registrul A nu reprezintă o cifră, dar dacă reprezintă, atunci forma în virgulă mobilă a cifrei devine 'ultima valoare' din stiva calculatorului.

```
2D22 STK-DIGIT  CALL  2D1B,NUMERIC          Este caracterul o cifră?
                RET   C                     Revenire dacă nu este în
                                            interval.
                SUB   +30                   Se înlocuieste codul cu cifra
                                            actuală.
```

THE 'STACK-A' SUBROUTINE (SUBRUTINA 'STACK-A')
Această subrutină dă forma în virgulă mobilă pentru valoarea binară absolută curentă continută în registrul A.

```
2D28 STACK-A    LD    C,A                   Se transferă valoarea în
                                            registrul C.
                LD    B,+00                 Se sterge registrul B.
```

THE 'STACK-BC' SUBROUTINE (SUBRUTINA 'STACK-BC')
Această subrutină dă forma în virgulă mobilă pentru valoarea binară absolută curentă continută în registrul BC.

Forma utilizată în aceasta si deci la fel de bine în două subrutine anterioare este aceea utilizată în Spectrum pentru numere întregi mici n, unde -65535 <= n <= 65535. Octetii unu si cinci sînt zero; al treilea si patrulea

octet sînt octetii cei mai putin si cei mai semnificativi octeti ai celor 16 biti de întreg n în complement fatá de doi (dacá n este negativ, acesti doi octeti vor contine 65536+n); si al doilea octet este un octet de semn, 00 pentru + si FF pentru '-'.

```
2D2B STACK-BC   LD   IY,+5C3A          Reinitializare IY pe ERR-NR.
                XOR  A                 Se sterge registrul A.
                LD   E,A               Si registrul E, pentru a
                                       indica '+'.
                LD   D,C               Se copiazá octetul cel mai
                                       putin semnificativ în D.
                LD   C,B               Si octetul cel mai
                                       semnificativ în C.
                LD   B,A               Se sterge registrul B.
                CALL 2AB6,STK-STORE    Acum se stocheazá numárul.
                RST  0028,FP-CALC      HL va indica STKEND - 5.
                DEFB +38,end-calc
                AND  A                 Se sterge fanionul de
                                       transport.
                RET                    Sfîrsit.
```

THE 'INTEGER TO FLOATING-POINT' SUBROUTINE (SUBRUTINA 'INTREG IN VIRGULA MOBILA')

Aceastá subrutiná revine cu 'ultima valoare' din stiva calculatorului ca rezultat al al conversiei unui întreg dintr-o linie BASIC, adicá a pártii întregi a unui número zecimal sau a unui número de linie în forma sa în virgulá mobilá.

Prin apelarea repetatá a lui CH-ADD+1 se aduce pe rînd fiecare cifrá a întregului. Se iese atunci cînd s-a adus un cod care nu reprezintá o cifrá.

```
2D3B INT-TO-FP  PUSH AF                Se salveazá prima cifrá în A.
                RST  0028,FP-CALC      Utilizare calculator.
                DEFB +A0,stk-zero      Acum 'ultima valoare' este
                                       zero.
                DEFB +38,end-calc
                POP  AF                Se readuce prima cifrá.
```

Acum se introduce o buclá. Atîta timp cît codul reprezintá o cifrá se gáseste si se stocheazá sub 'ultima valoare'. Apoi 'ultima valoare' este multiplicatá cu 10 zecimal si adunatá cu 'cifra' pentru a forma noua 'ultima valoare' care este transportatá înapoi la începutul buclei.

```
2D40 NXT-DGT-2  CALL 2D22,STK-DIGIT    Dacá codul reprezintá o
                RET  C                 cifrá, atunci se stocheazá
                                       forma în virgulá mobilá.
                RST  0028,FP-CALC      Se utilizeazá calculatorul.
                DEFB +01,exchange      'Cifra' trece sub 'ultima
                                       valoare'.
                DEFB +A4,stk-ten       Se defineste zecimalul 10.
                DEFB +04,multiply      'Ultima valoare' = 'ultima
                                       valoare' * 10
                DEFB +0F,addition      'Ultima valoare' = 'ultima
                                       valoare' + 'cifra'
                DEFB +38,end-calc
                CALL 0074,CH-ADD+1     Urmátorul cod trece în
                                       registrul A.
                JR   2D40,NXT-DGT-2    Salt înapoi cu acest cod.
```

THE ARITHMETIC ROUTINES (RUTINELE ARITMETICE)

THE 'E-FORMAT TO FLOATING-POINT' SUBROUTINE (SUBRUTINA 'E - FORMAT IN VIRGULA MOBILA')

(Deplasament 3C - vezi CALCULATE după: 'e-to-fp')

Această subrutină dă 'ultima valoare' din vîrful stivei calculatorului care este rezultatul conversiei numărului în forma xEm, unde m este un întreg pozitiv sau negativ. Subrutina este introdusă cu x în vîrful stivei calculatorului și cu m în registrul A.

Metoda care se foloseste este de a afla valoarea absolută a lui m, numită p, si de a înmulti sau a împărti x cu 10^p în concordantă cu faptul că m este pozitiv sau negativ.

Pentru a realiza acest lucru, p este deplasat la dreapta pînă cînd devine zero, si x este înmultit sau împărtit cu 10^(2^n) pentru fiecare bit setat b(n) al p. Pînă cînd p nu este niciodată mai mare decît 39 zecimal, bitii 6 si 7 ai lui p nu vor fi în mod normal setati.

```
2D4F E-TO-FP      RLCA                          Se testează semnul lui m
                  RLCA                          rotind bitul 7 din A în
                                                transport fără a-l schimba pe
                                                A.
                  JR    NC,2D55,E-SAVE          Salt dacă m este pozitiv.
                  CPL                           Se face negarea lui m din A
                  INC   A                       fără a distruge fanionul de
                                                transport.
2D55 E-SAVE       PUSH  AF                      Se salvează m în A pentru
                                                scurt timp.
                  LD    HL,+5C92                Este MEMBOT: un fanion de
                  CALL  350B,FP-0/1             semn este acum stocat în
                                                primul octet al lui mem-0,
                                                adică 0 pentru '+' si '1'
                                                pentru '-'.
                  RST   0028,FP-CALC            Stiva contine x.
                  DEFB  +A4,stk-ten             x,10(zecimal).
                  DEFB  +38,end-calc            x,10
                  POP   AF                      Se readuce m în A.
2D60 E-LOOP       SRL   A                       In buclă, se deplasează afară
                                                următorul bit al lui m,
                                                modificînd corespunzător
                                                fanionul zero si de
                                                transport;
                  JR    NC,2D71,E-TST-END       salt dacă transportul este
                                                resetat.
                  PUSH  AF                      Se salvează restul din m si
                                                fanioanele.
                  RST   0028,FP-CALC            Stiva contine x' si 10^(2^n),
                                                unde x' este o etapă
                                                intermediară în multiplicarea
                                                lui x cu 10^m, si n=0,1,2,3,4
                                                sau 5.
                  DEFB  +C1,st-mem-1            (10^(2^n) este copiat în
                                                mem-1.)
                  DEFB  +E0,get-mem-0           x',10^(2^n),(1/0)
                  DEFB  +00,jump-true           x',10^(2^n)
                  DEFB  +04,to E-DIVSN          x',10^(2^n)
                  DEFB  +04,multiply            x'*10^(2^n) = x"
                  DEFB  +33,jump                x"
                  DEFB  +02,to E-FETCH          x"
2D6D E-DIVSN      DEFB  +05,division            x'/10^(2^n) = x"  (x" este
                                                N'*10^(2^n) sau x'/10^(2^n)
                                                în concordantă cu faptul că m
                                                este '+' sau '-')
2D6E E-FETCH      DEFB  +E1,get-mem-1           x",10^(2^n)
                  DEFB  +38,end-calc            x",10^(2^n)
                  POP   AF                      Se reface restul din m în A,
                                                si fanioanele.
2D71 E-TST-END    JR    Z,2D7B,E-END            Salt dacă m a fost redus la
                                                zero.
                  PUSH  AF                      Se salvează restul lui m în A
                  RST   0028FP-CALC             x",10^(2^n)
                  DEFB  +31,duplicate           x",10^(2^n),10^(2^n)
                  DEFB  +04,multiply            x",10^(2^(n+1))
                  DEFB  +38,end-calc            x",10^(2^(n+1))
                  POP   AF                      Se readuce restul lui m în A.
                  JR    2D60,E-LOOP             Salt înapoi pentru toti bitii
                                                lui m.
2D7B E-END        RST   0028,FP-CALC            Se foloseste calculatorul
                  DEFB  +02,delete              pentru a sterge puterea
```

```
                DEFB  +38,end-calc        finală a lui 10 la care s-a
                RET                       ajuns, lăsînd în stivă
                                          'ultima valoare' x'10^m.
```

THE 'INT-FETCH' SUBROUTINE (SUBRUTINA 'INT-FETCH')
Această subrutină colectează în DE un număr întreg n (-65535 <= n <= 65535) din locatia adresată de HL; aceasta înseamnă că n este în mod normal primul (sau al doilea) număr din vîrful stivei calculatoului; dar HL poate de asemenea accesa (prin schimbare cu DE) un număr care a fost sters din stivă.
Subrutina nu sterge ea însăsi numărul din stivă sau din memorie; ea revine cu HL indicînd al patrulea octet al numărului în pozitia sa initială.

```
2D7F INT-FETCH  INC   HL                  Se indică octetul de semn al
                                          numărului.
                LD    C,(HL)              Se copiază octetul de semn în
                                          C.
```

Mecanismul următor va complementa față de 2 numărul dacă este negativ (C este FF) dar îl va lăsa neschimbat dacă este pozitiv (C este 00).

```
                INC   HL                  Se indică octetul cel mai
                                          putin semnificativ.
                LD    A,(HL)              Se colectează octetul în A.
                XOR   C                   Dacă este negativ se face
                                          complementul său față de 1.
                SUB   C                   Aceasta adună 1 pentru numere
                                          negative; se setează
                                          transportul numai dacă
                                          octetul a fost 0.
                LD    E,A                 Acum se pune octetul cel mai
                                          putin semnificativ în E.
                INC   HL                  Se indică octetul cel mai
                                          semnificativ.
                LD    A,(HL)              Acesta se coleactează în A.
                ADC   A,C                 Se termină complementarea
                XOR   C                   față de doi în cazul unui
                                          număr negativ; de notat că
                                          transportul este întodeauna
                                          lăsat resetat.
                LD    D,A                 Acum se trece octetul cel mai
                                          semnificant în D.
                RET                       Sfîrsit.
```

THE 'INT-STORE' SUBROUTINE (SUBRUTINA 'INT-STORE')
Această subrutină stochează un număr întreg mic n (-65535<= n <=65535) în locatia adresată de HL si următoarele patru locatii; aceasta înseamnă că se înlocuieste primul (sau al doilea) număr din vîrful stivei calculatorului.
Subrutina revine cu HL indicînd primul octet al lui n din stivă.

```
2D8C P-INT-STO  LD    C,+00               Acest punct de intrare poate
                                          stoca un număr stiut ca fiind
                                          pozitiv.
2D8E INT-STORE  PUSH  HL                  Este salvat indicatorul
                                          primei locatii.
                LD    (HL),+00            Primul octet este setat pe
                                          zero.
                INC   HL                  Se indică a doua locatie.
                LD    (HL),C              Se introduce octetul de semn.
```

Acelasi mecanism este folosit acum ca în 'INT-FETCH' pentru a complementa față de doi numere negative. Aceasta este necesar, de exemplu, înainte si după înmultirea numerelor întregi mici. Adunarea este oricum efectuată fără vreo complementare față de doi înainte sau după aceea.

```
                INC   HL                  Se indică a treia locatie.
                LD    A,E                 Se colectează cel mai putin
                                          semnificativ octet.
                XOR   C                   Dacă numărul este negativ se
                SUB   C                   face complementul față de
                                          doi.
                LD    (HL),A              Se stochează octetul.
                INC   HL                  Se indică a patra locatie.
                LD    A,D                 Se colectează octetul cel mai
                                          semnificativ.
                ADC   A,C                 Dacă numărul este negativ se
                XOR   C                   face complementul față de
                                          doi.
                LD    (HL),A              Se stochează octetul.
                INC   HL                  Se indică a cincea locatie.
```

```
                LD    (HL),+00        Al cincelea octet este setat
                                      pe zero.
                POP   HL              Se revine cu HL indicînd
                RET                   primul octet al lui n din
                                      stivă.
```

THE 'FLOATING-POINT TO BC' SUBROUTINE (SUBRUTINA 'VIRGULA MOBILA IN BC')
Această subrutină este apelată din patru locuri diferite pentru diferite situatii si este folosită pentru a comprima 'ultima valoare' în virgulă mobilă în registrul pereche BC.
Dacă rezultatul este prea mare, adică mai mare decît 65535 zecimal, atunci subrutina revine cu fanionul de transport setat. Dacă 'ultima valoare' este negativă atunci fanionul zero este resetat.
Octetul cel mai putin semnificativ al rezultatului este copiat în registrul A.

```
2DA2 FP-TO-BC   RST   0028,FP-CALC     Se utilizează calculatorul
                DEFB  +38,end-calc      pentru a face ca HL să indice
                                        STKEND - 5.
                LD    A,(HL)            Se colectează octetul
                AND   A                 exponent al 'ultimei valori';
                JR    Z,2DAD,FP-DELETE  salt dacă este zero, el
                                        indicînd un 'număr întreg
                                        mic'.
                RST   0028,FP-CALC      Acum se utilizează
                DEFB  +A2,stk-half      calculatorul pentru a rotunji
                DEFB  +0F,addition      'ultima valoare' la cel mai
                DEFB  +27,int           apropiat întreg, care de
                DEFB  +38,end-calc      asemenea se schimbă în forma
                                        'întreagă cea mai mică' din
                                        stiva calculatorului dacă
                                        acest lucru este posibil,
                                        adică - 65535.5<=x<=65535.5.
2DAD FP-DELETE  RST   0028,FP-CALC      Se foloseste calculatorul
                DEFB  +02,delete        pentru a sterge întregul din
                DEFB  +38,end-calc      stivă; DE încă îl indică în
                                        memorie (la STKEND).
                PUSH  HL                Se salvează ambii indicatori.
                PUSH  DE
                EX    DE,HL             Acum HL indică numărul.
                LD    B,(HL)            Se copiază primul octet în B.
                CALL  2D7F,INT-FETCH    Se copiază octetii 2, 3 si 4
                                        în C, E si D.
                XOR   A                 Se sterge registrul A.
                SUB   B                 Aceasta setează transportul
                                        numai dacă B este zero.
                BIT   7,C               Aceasta setează fanionul zero
                                        numai dacă numărul este
                                        pozitiv (NZ denotă negativ).
                LD    B,D               Se copiază octetul cel mai
                                        semnificativ în B.
                LD    C,E               Si octetul cel mai putin
                                        semnificativ în C.
                LD    A,E               Octetul cel mai putin
                                        semnificativ este copiat si
                                        în A.
                POP   DE                Se readuc indicatorii stivă.
                POP   HL
                RET                     Sfîrsit.
```

THE 'LOG (2^A)' SUBROUTINE (SUBRUTINA 'LOG (2 ^ A)
Această subrutină este apelată de subrutina 'PRINT-FP' pentru a calcula numărul aproximativ de cifre dinaintea punctului zecimal în x, numărul de tipărit, sau, dacă nu sînt cifre înaintea punctului zecimal, atunci numărul aproximativ de zerouri de fond după punctul zecimal. Ea este introdusă cu registrul A continînd e', exponentul real al lui x, sau e'-2, si calculează z=log în baza 10 din (2^A). Apoi îl setează pe A egal cu ABS INT (z+0.5), cum se cere, folosind în acest scop FP-TO-A.

```
2DC1 LOG (2 A)  LD    D,A               Intregul A este stocat, fie
                RLA                     ca 00 00 A 00 00 (pentru A
                SBC   A,A               pozitiv) sau ca 00 FF A FF 00
                                        (pentru A negativ).
                LD    E,A               Acesti octeti sînt mai întîi
                LD    C,A               încărcati în A, E, D, C, B si
                XOR   A                 apoi STKEND este apelat
                LD    B,A               pentru a pune numărul în
                                        stiva calculatorului.
                CALL  2AB6,STK-STORE
```

```
                RST   0028,FP-CALC          Utilizare calculator.
                DEFB  +34,stk-data          Acum este stocat log 2 în
                                            baza 10.
                DEFB  +EF,exponent+7F       Acum stiva contine A, log 2.
                DEFB  +1A,+20,+9A,+85
                DEFB  +04,multiply          A*log 2, adică log (2^A).
                DEFB  +27,int               INT log (2^A)
                DEFB  +38,end-calc
```

Subrutina continuă în FP-TO-A pentru a completa calculul.

THE 'FLOATING-POINT TO A' SUBROUTINE (SUBRUTINA 'A IN VIRGULA MOBILA')
Această subrutină scurtă dar vitală este apelată de cel puțin 8 ori pentru diferite scopuri. Ea folosește penultima subrutină, FP-TO-BC, pentru a aduce 'ultima valoare' în registrul A unde este posibil. De aceea ea testează dacă modulul numărului rotunjit este mai mare decît 255 și dacă a realizat revenirea subrutinei cu fanionul de transport setat. Altfel ea revine cu modulul numărului, rotunjit la cel mai apropiat întreg, în registrul A, și fanionul zero setat implică faptul că numărul a fost pozitiv, sau resetat, ceea ce implică faptul că a fost negativ.

```
2DD5 FP-TO-A    CALL  2DA2,FP-TO-BC         Se comprimă 'ulima valoare'
                                            în BC.
                RET   C                     Se revine dacă s-a iesit deja
                                            din interval.
                PUSH  AF                    Se salvează rezultatul și
                                            fanioanele.
                DEC   B                     Din nou se va iesi din
                INC   B                     interval dacă registrul B nu
                                            contine zero.
                JR    Z,2DE1,FP-A-END       Salt dacă este în interval.
                POP   AF                    Se aduce rezultatul și
                                            fanioanele.
                SCF                         Se semnalează că rezultatul
                                            este afară din interval.
                RET                         Sfîrșit - nereusită.
2DE1 FP-A-END   POP   AF                    Se aduce rezultatul și
                                            fanioanele.
                RET                         Sfîrșit - reusită.
```

THE 'PRINT A FLOATING-POINT NUMBER' SUBROUTINE (SUBRUTINA 'TIPARIRE NUMAR IN VIRGULA MOBILA)
Această subrutină este apelată de rutina de comandă PRINT la 2039 și de STR$ la 3630, ea convertind într-un șir numărul așa cum va fi tipărit. Subrutina tipărește x, 'ultima valoare' din stiva calculatorului. Formatul pentru tipărire nu ocupă niciodată mai mult de 14 spații.

Cele mai semnificative 8 cifre ale lui x, rotunjit corect, sînt stocate într-un buffer de tipărire ad-hoc în mem-3 și mem-4. Numerele mici, numeric mai mici decît 1, și numerele mari, numeric mai mari decît 2^27, sînt prelucrate separat. Numărul anterior este înmultit cu 10^n, unde n este numărul aproximat de zerouri de fond după punctul zecimal, pe cînd cel recent se împarte prin 10^(n-7), unde n este numărul aproximat de cifre înainte de punctul zecimal. Aceasta aduce toate numerele într-un interval de mijloc, și numărul de cifre cerute înainte punctului zecimal este construit în al doilea octet al lui mem-5. In final se efectuează tipărirea, folsind formatul E dacă sînt mai mult de 8 cifre înainte de punctul zecimal, sau, pentru numere mici, mai mult de 4 zerouri de fond după punctul zecimal.

Următorul program arată ordinea de tipărire a formelor:

```
10 FOR a = -11 TO 12: PRINT SGN a*9^a.: NEXT a
```

i. In primul rînd semnul lui x tine cont de:

Dacă x este negativ, subrutina sare la PF-NEGATIVE, ia ABS x și tipărește semnul minus.

Dacă x este zero, x este sters din stiva calculatorului, se tipărește un zero și se revine din subrutină.

Dacă x este pozitiv, subrutina continuă.

```
2DE3 PRINT-FP   RST   0028,FP-CALC          Utilizare calculator.
                DEFB  +31,duplicate         x,x
                DEFB  +36,less-0            x,(1/0) Valoarea logică a lui
                                            x.
                DEFB  +00,jump-true         x
                DEFB  +0B,toPF-NEGTVE       x
                DEFB  +31,duplicate         x,x
                DEFB  +37,greater-0         x,(1/0) Valoarea logică a lui
                                            x.
                DEFB  +00,jump-true         x
                DEFB  +0D,to PF-POSTVE      x De aici încolo x'=ABS x
                DEFB  +02,delete            -
```

```
                DEFB  +38,end-calc        ;
                LD    A,+30               Se introduce codul caracter
                                          pentru '0'.
                RST   0010,PRINT-A-1      Se tipăreste '0'.
                RET                       Se  termină  cînd  'ultima
                                          valoare' este zero.
2DF2 PF-NEGTVE  DEFB  +2A,abs             x'   x'=ABS x
                DEFB  +38,end-calc        x'
                LD    A,+2D               Se introduce codul caracter
                                          pentru '-'.
                RST   0010,PRINT-A-1      se tipăreste '-'.
                RST   0028,FP-CALC        Se  foloseste  din  nou
                                          calculatorul.
2DF8 PF-POSTVE  DEFB  +A0,stk-zero        Cei  15  octeti ai lui mem-3,
                DEFB  +C3,st-mem-3        mem-4  si  mem-5  sînt  acum
                DEFB  +C4,st-mem-4        initializati pe zero pentru a
                DEFB  +C5,st-mem-5        fi folositi ca buffer pentru
                                          tipărire si două contoare.
                DEFB  +02,delete          Stiva  este  stearsă,  cu
                                          exceptia lui x'.
                DEFB  +38,end-calc        x'
                EXX                       H'L',  care  este  folosit
                PUSH  HL                  pentru  a  tine deplasamentul
                EXX                       calculatorului  (de  exemplu
                                          'STR$') este salvat în stiva
                                          masinii.
```

ii. Acesta este începutul unei bucle care prelucrează numere mari. Oricum fiecare număr x este mai întîi despărtit în partea sa întreagă i si în partea sa fractionară f. Dacă i este un întreg mic, adică dacă -65535<= i <= 65535, el este stocat în DE pentru inserarea în bufferul de tipărire.

```
2E01 PF-LOOP    RST   0028,FP-CALC        Utilizare calculator.
                DEFB  +31,duplicate       x',x'
                DEFB  +27,int             x', INT (x')=i
                DEFB  +C2,st-mem-2        (i este stocat în mem-2)
                DEFB  +03,subtract        x'-i=f
                DEFB  +E2,get-mem-2       f,i
                DEFB  +01,exchange        i,f
                DEFB  +C2,st-mem-2        (f este stocat în mem-2)
                DEFB  +02,delete          i
                DEFB  +38,end-calc        i
                LD    A,(HL)              Este i un întreg mic (primul
                AND   A                   octet zero), adică ABS i <=
                                          65535?
                JR    NZ,2E56,PF-LARGE    Salt dacă nu este.
                CALL  2D7F,INT-FETCH      i este copiat în DE (i, ca si
                                          x', >=0)
                LD    B,+10               B este setat pentru a număra
                                          16 biti.
                LD    A,D                 D  este  copiat în A pentru a
                                          fi testat:
                AND   A                   este zero?
                JR    NZ,2E1E,PF-SAVE     Salt dacă nu este zero.
                OR    E                   Acum se testează E.
                JR    Z,2E24,PF-SMALL     Salt dacă E este zero: x este
                                          o fractie curată.
                LD    D,E                 Se mută E în D si se setează
                LD    B,+08               B  pentru  8  biti:  D a fost
                                          zero iar E nu a fost.
2E1E PF-SAVE    PUSH  DE                  Se transferă DE în D'E', prin
                EXX                       stiva  masinii,  pentru  a fi
                POP   DE                  mutati  în  bufferul  de
                EXX                       tipărire la PF-BITS.
                JR    2E7B,PF-BITS        Salt înainte.
```

iii. Fractiile curate sînt înmultite cu 10^n, unde n este numărul aproximat de zerouri de fond după punctul zecimal; si -n este adunat la al doilea octet al lui mem-5, care contine numărul de cifre necesare înainte de punctul zecimal; un număr negativ aici este indicat prin zerouri de fond după punctul zecimal.

```
2E24 PF-SMALL   RST   0028,FP-CALC        i (i= zero aici)
                DEFB  +E2,get-mem-2       i,f
                DEFB  +38,end-calc        i,f
```

De notat că stiva acum este nesimetrică. Este necesar un octet aditional 'DEFB +02,delete' la 2E25, imediat după RST 0028. Acum, o expresie ca "2" +STR$ 0.5, este evaluată incorect ca 0.5; zeroul din stînga din stivă înlocuieste "2" si este tratată ca un sir invalid. Analog toate comparatiile sirului pot

produce valori incorecte dacă al doilea sir ia forma STR$ x, unde x este numeric mai mic decît 1, de exemplu expresia "50"<STR$ 0.1 produce valoarea logică "adevărat"; din nou "" este folosit în loc de "50".

```
                LD    A,(HL)          Octetul exponent al lui f
                                      este copiat în A.
                SUB   +7E             A devine e-126 zecimal, adică
                                      e'+2, unde e' este exponentul
                                      real al lui f.
                CALL  2DC1,LOG(2^A)   Este realizată construcţia
                                      A = ABS INT (LOG (2^A)) (LOG
                                      este în baza 10); asta
                                      înseamnă A=n, unde n este
                LD    D,A             copiat din A în D.
                LD    A,(mem-5-2nd)   Contorul curent este colectat
                SUB   D               din al doilea octet al lui
                LD    (mem-5-2nd),A   mem-5 si n este scăzut din el
                LD    A,D             n este copiat din D în A.
                CALL  2D4F,E-TO-FP    y=f*10^n este format si
                                      stocat.
                RST   0028,FP-CALC    i,y
                DEFB  +31,duplicate   i,y,y
                DEFB  +27,int         i,y,INT (y) = i2
                DEFB  +C1,st-mem-1    (i2 este copiat în mem-1)
                DEFB  +03,subtract    i,y-i2
                DEFB  +E1,get-mem-1   i,y-i2,i2
                DEFB  +38,end-calc    i,f2,i2 (f2 = y - i2)
                CALL  2DD5,FP-TO-A    i2 este transferat din stivă
                                      în A.
                PUSH  HL              Indicatorul lui f2 este
                                      salvat.
                LD    (mem-3-1st),A   i2 este stocat în primul
                                      octet al lui mem-3: o cifră
                                      pentru tipărire.
                DEC   A               i2 nu va număra cifra pentru
                RLA                   tipărire dacă ea este zero; A
                                      este modificat astfel încît
                SBC   A,A             zero va produce zero dar
                INC   A               cifră diferită de zero v
                                      produce 1.
                LD    HL,+5CAB        Zero sau unu este inserat î
                LD    (HL),A          primul octet al lui mem-
                INC   HL              (numărul de cifre pentr
                ADD   A,(HL)          tipărire) si adunat cu a
                                      doilea octet al lui mem-5.
                                      (numărul de cifre dinainte de
                                      punctul zecimal).
                POP   HL              Se readuce indicatorul lui
                                      f2.
                JP    2ECF,PF-FRACTN  Salt pentru a stoca f2 în
                                      buffer (acum HL indică f2,
                                      iar DE indică i2).
```

iv. Numerele mai mari decît 2 ^ 27 sînt multiplicate similar de 2 ^ (-n+7), reducînd numărul cifrelor dinainte de punctul zecimal la 8, si bucla reintră la PF-LOOP.

```
2E56 PF-LARGE   SUB   +80             e-80 hex=e', exponentul real
                                      al lui i.
                CP    +1C             Este e' mai mic decît 28
                                      zecimal?
                JR    C,2E6F,PF-MEDIUM Salt dacă este mai mic.
                CALL  2DC1,LOG(2^A)   n este format în A.
                SUB   +07             Si redus la n-7.
                LD    B,A             Apoi este copiat în B.
                LD    HL,+5CAC        n-7 este adăugat la al doilea
                ADD   A,(HL)          octet al lui mem-5, numărul
                LD    (HL),A          de cifre cerut înainte de
                LD    A,B             punctul zecimal al lui x.
                                      Apoi i este înmultit cu
                                      10^(-n+7).
                NEG                   Aceasta îl va da într-un
                CALL  2D4F,E-TO-FP    interval de mijloc pentru
                                      tipărire.
                JR    2E01,PF-LOOP    Se ciclează din nou bucla
                                      pentru a prelucra acum
                                      numărul mărime-medie.
```

v. Partea întreagă a lui x este acum stocată în bufferul de tipărire în mem-3

si mem-4.

```
2E6F PF-MEDIUM   EX    DE,HL          Acum DE îl indică pe i, HL pe
                                      e.
                 CALL  2FBA,FETCH-TWO Mantisa lui i este acum în
                                      D',E',D,E.
                 EXX                  Se schimbă registrii.
                 SET   7,D            Bitul 7 numeric adevărat în
                                      D'.
                 LD    A,L            octetul exponent e al lui i
                                      în A.
                 EXX                  Se aduc înapoi registrii
                                      principali.
                 SUB   +80            Exponentul real e'=e-80 hex
                                      în A.
                 LD    B,A            Acesta dă contorul de bit
                                      care s-a cerut.
```

De notat că în cazul în care i este un număr întreg mic (mai mic decît 65536) se reintră aici.

```
2E7B PF-BITS     SLA   E              Acum mantisa lui i se roteste
                 RL    D              la stînga si toti bitii lui i
                 EXX                  sînt aceia deplasati în mem-4
                 RL    E              si fiecare octet al lui mem-4
                                      este ajustat zecimal la
                                      fiecare deplasare.
                 RL    D              Toti cei patru octeti ai lui
                                      i.
                 EXX                  Intoarcere la registrii
                                      principali.
                 LD    HL,+5CAA       Se pune în HL adresa celui de
                 LD    C,+05          al cincelea octet al lui
                                      mem-4; contor pentru 5 octeti
                                      în C.
2E8A PF-BYTES    LD    A,(HL)         Se aduce octetul din mem-4.
                 ADC   A,A            Se deplasează la stînga,
                                      luînd noul bit.
                 DAA                  Ajustarea zecimală a
                                      octetului.
                 LD    (HL),A         Este readus în mem-4.
                 DEC   HL             Se indică următorul octet al
                                      lui mem-4.
                 DEC   C              Se decrementează contorul de
                                      biti cu unu.
                 JR    NZ,2E8A,PF-BYTES Salt pentru fiecare octet al
                                      lui mem-4.
                 DJNZ,PF-BITES        Salt pentru fiecare bit al
                                      lui INT (x).
```

Ajustînd zecimal fiecare octet al lui mem-4 se obtin două cifre zecimale per octet, fiind cel mult nouă cifre. Cifrele vor fi acum reîmpachetate, una într-un octet, în mem-3 si mem-4, folosind instructia RLD.

```
                 XOR   A              A este sters pentru primirea
                                      cifrelor.
                 LD    HL,+5CA6       Adresa sursă: primul octet al
                                      lui mem-4.
                 LD    DE,+5CA1       Destinatia: primul octet al
                                      lui mem-3.
                 LD    B,+09          Sînt cel mult 9 cifre.
                 RLD                  Bucata din stînga al lui
                                      mem-4 este înlăturată.
                 LD    C,+FF          FF în C va semnala un zero de
                                      fond, iar 00 va semnala un
                                      un nonzero de fond.
2EA1 PF-DIGITS   RLD                  Partea stîngă a lui (HL) în A
                                      iar partea dreaptă a lui (HL)
                                      la stînga.
                 JR    NZ,2EA9,PF-INSERT Salt dacă cifra din A nu este
                                      zero.
                 DEC   C              Testare zero de fond;
                 INC   C              aceasta va da acum resetarea
                                      lui zero.
                 JR    NZ,2EB3,PF-TEST-2 Salt dacă a fost un zero de
                                      de fond.
2EA9 PF-INSERT   LD    (DE),A         Acum se introduce cifra.
                 INC   DE             Se indică următoarea
                                      indicatie.
```

```
                INC   (mem-5-1st)       Incă o cifră de tipărit, si
                INC   (mem-5-2nd)       încă una după punctul zecimal
                LD    C,+00             Se schimbă fanionul din zero
                                        de fond în alt zero.
2EB3 PF-TEST-2  BIT   0,B               Indicatorul sursă trebuie
                JR    Z,2EB8,PF-ALL-9   incrementat la fiecare a doua
                INC   HL                trecere prin buclă, cînd B
                                        este impar.
2EB8 PF-ALL-9   DJNZ  2EA1,PF-DIGITS    Salt înapoi pentru toate cele
                                        9 cifre.
                LD    A,(mem-5-1st)     Se aduce contorul: există
                SUB   +09               acolo cele 9 cifre care
                JR    C,2ECB,PF-MORE    exclud zerourile de fond?
                DEC   (mem-5-1st)       Pregătire ciclare: se reduce
                                        contorul la 8.
                LD    A,+04             Se compară a 9-a cifră din
                CP    (mem-4-4th)       octetul 4 al lui mem-4, cu 4
                                        pentru a seta transportul
                                        pentru rotunjire.
                JR    2F0C,PF-ROUND     Salt mai departe pentru
                                        rotunjire.
2ECB PF-MORE    RST   0028,FP-CALC      Se foloseste din nou
                                        calculatorul.
                DEFB  +02,delete        (i este acum sters.)
                DEFB  +E2,get-mem-2     f
                DEFB  +38,end-calc      f
```

vi. Partea fractionară a lui x este acum stocată în bufferul de tipărire.

```
2ECF PF-FRACTN  EX    DE,HL             Acum DE îl indică pe f.
                CALL  2FBA,FETCH-TWO    Mantisa lui f este acum în
                                        D',E',D,E.
                EXX                     Schimbare registrii.
                LD    A,+80             Exponentul lui f este redus
                SUB   L                 la zero, rotind bitii lui f
                LD    L,+00             cu 80 hex. - e locuri la
                                        dreapta, unde L' contine e.
                SET   7,D               Bitul numeric real în bitul 7
                                        din D'.
                EXX                     Se readuc registrii
                                        principali.
                CALL  2FDD,SHIFT-FP     Acum se face deplasarea.
2EDF PF-FRN-LP  LD    A,(mem-5-1st)     Se aduce contorul cifrelor.
                CP    +08               Sînt deja 8 cifre?
                JR    C,2EEC,PR-FR-DGT  Dacă nu, salt înainte.
                EXX                     Dacă sînt 8 cifre, se
                RL    D                 foloseste f pentru a-l
                                        rotunji pe i, rotind D' la
                                        stînga pentru a seta
                                        transportul.
                EXX                     Se readuc registrii
                JR    2F0C,PF-ROUND     principali si salt în fată
                                        pentru rotunjire.
2EEC PF-FR-DGT  LD    BC,+0200          Initial zero în C, contorul
                                        de 2 în B.
2EEF PF-FR-EXX  LD    A,E               D'E'DE este înmultit cu 10 în
                CALL  2F8B,CA=10*A+C    2 etape, mai întîi DE apoi
                LD    E,A               D'E', fiecare octet cu octet
                LD    A,D               în 2 etape, si partea
                CALL  2F8B,CA=10*A+C    întreagă a rezultatului este
                LD    D,A               obtinută în C pentru a fi
                                        trecută în bufferul de
                                        tipărire.
                PUSH  BC                Contorul si rezultatul
                EXX                     alternează între BC si B'C'.
                POP   BC
                DJNZ  2EEF,PF-FR-EXX    Privire înapoi de-a lungul
                                        schimbării registrilor.
                LD    HL,+CA1           inceputul - primul octet al
                                        lui mem-3.
                LD    A,C               Rezultatul este pus în A
                                        pentru stocare.
                LD    C,(mem-5-1st)     Contorul cifrelor de pînă
                                        acum din număr în C.
                ADD   HL,BC             Este adresat primul octet gol
                LD    (HL),A            Se stochează următoarea
                                        cifră.
                INC   (mem-5-1st)       Se măreste contorul cifrelor.
                JR    2EDF,PF-FRN-LP    Se ciclează pînă sînt opt
```

cifre.

vii. Cifrele stocate în bufferul de tipărire sînt rotunjite la maxim 8 cifre pentru tipărire.

```
2F0C PF-ROUND    PUSH  AF              Se salvează fanionul de
                                       transport pentru rotunjire.
                 LD    HL,+5CA1        Adresa de bază a numărului:
                                       mem-3, octetul 1.
                 LD    C,(mem-5-1st)   Deplasamentul (numărul de
                 LD    B,+00           cifre din număr) în BC.
                 ADD   HL,BC           Se adresează ultimul octet al
                                       numărului.
                 LD    B,C             Se copiază C în B ca si
                                       contor.
                 POP   AF              Se readuce fanionul de
                                       transport.
2F18 PF-RND-LP   DEC   HL              Acesta este ultimul octet al
                                       numărului.
                 LD    A,(HL)          Se aduce octetul în A.
                 ADC   A,+00           Adunare în transport, adică
                                       rotunjire.
                 LD    (HL),A          Se stochează octetul rotunjit
                                       în buffer.
                 AND   A               Dacă octetul este 0 sau 10, B
                 JR    Z,2F25,PF-R-BACK va fi decrementat si zero din
                 CP    +0A             fina (sau 10) nu vor fi
                                       numărati pentru tipărire.
                 CCF                   Se resetează transportul
                                       pentru o cifră validă.
                 JR    NC,2F2D,PF-COUNT Salt dacă transportul este
                                       resetat.
2F25 PF-R-BACK   DJNZ  2F18,PF-RND-LP  Salt înapoi pentru mai multe
                                       rotunjiri sau mai multe
                                       zerouri în final.
                 LD    (HL),+01        Este o depăsire la stînga;
                 INC   B               este necesar aici un 1
                                       aditional.
                 INC   (mem-5-2nd)     Este de asemenea o cifră
                                       aditională înainte de punctul
                                       zecimal.
2F2D PF-COUNT    LD    (mem-5-1st),B   Acum B setează contorul
                                       cifrelor ce vor fi tipărite
                                       (zerourile finale nu vor fi
                                       tipărite).
                 RST   0028,FP-CALC    f trebuie sters.
                 DEFB  +02,delete      -
                 DEFB  +38,end-calc    -
                 EXX
                 POP   HL              Deplasamentul calculatorului
                                       care a fost salvat în stivă
                                       este readus în H'L'.
                 EXX
```

viii. Acum se poate tipări numărul. Mai întîi C va fi setat astfel încît să conțină numărul cifrelor ce trebuie tipărite, necontorizînd zerourile finale, în timp ce B va conține numărul cifrelor cerute înainte de punctul zecimal.

```
                 LD    BC,(mem-5-1st)  Contoarele sînt setate.
                 LD    HL,+5CA1        Începutul cifrelor.
                 LD    A,B             Dacă cifrele cerute înainte
                 CP    +09             virgulă sînt mai mult de 9,
                 JR    C,2F46,PF-NOT-E sau mai putin de 4, atunci
                                       este necesar formatul E.
                 CP    +FC             Mai putin de 4 înseamnă mai
                 JR    C,2F6C,PF-E-FRMT mult de 4 zerouri de fond
                                       după virgulă.
                 AND   A               Nu există cifre înainte de
                 CALL  Z,15EF,OUT-CODE virgulă? Dacă da, se
                                       tipăreste un zero initial.
```

Următorul punct de intrare este de asemenea folosit pentru a tipări cifrele necesare tipărite în formatul E.

```
2F4A PF-E-SBRN   XOR   A               Se începe prin setarea lui A
                                       pe zero.
                 SUB   B               Se scade B: minus va însemna
                 JR    M,2F52,PF-OUT-LP că există cifre înainte de
                                       virgulă; salt înainte pentru
                                       a le tipări.
```

```
                    LD    B,A           Acum se cere ca A să fie
                                        contor.
                    JR    2F5E,PF-DC-OUT Salt înainte pentru a tipări
                                        partea zecimală.
2F52 PF-OUT-LP      LD    A,C           Se copiază în A numărul
                    AND   A             cifrelor ce trebuie tipărite.
                    JR    Z,2F59,PF-OUT-DT Dacă A este zero, adică mai
                                        sînt zerouri finale de
                                        tipărit (B este diferit de
                                        zero), atunci se execută
                                        salt.
                    LD    A,(HL)        Se aduce o cifră din bufferul
                                        de tipărire.
                    INC   HL            Se indică următoarea cifră.
                    DEC   C             Se decrementează contorul cu
                                        unu.
2F59 PF-OUT-DT      CALL  15EF,OUT-CODE Se tipăreste cifra
                                        corespunzătoare.
                    DJNZ  2F52,PF-OUT-LP Se ciclează înapoi pînă cînd
                                        B este zero.
2F5E PF-DC-OUT      LD    A,C           Este momentul să se
                    AND   A             tipărească virgula, numai
                    RET   Z             dacă C este acum zero; în
                                        acest caz, se revine -
                                        sfîrsit.
                    INC   B             Se adună 1 la B - se include
                                        punctul zecimal.
                    LD    A,+2E         Se pune codul pentru '.' în
                                        A.
2F64 PF-DEC-OS      RST   0010,PRINT-A-1 Se tipăreste '.'.
                    LD    A,+30         Se introduce codul caracter
                                        pentru '0'.
                    DJNZ  2F64,PF-DEC-OS Se ciclează înapoi pentru a
                                        tipări toate zerourile
                                        necesare.
                    LD    B,C           Se setează contorul pentru
                                        toate cifrele rămase.
                    JR    2F52,PF-OUT-LP Salt înapoi pentru a le
                                        tipări.
2F6C PF-E-FRMT      LD    D,B           Contorul cifrelor este copiat
                                        în D.
                    DEC   D             El este decrementat pentru a
                                        da exponentul.
                    LD    B,+01         In format E se cere o cifră
                                        înainte de punctul zecimal.
                    CALL  2F4A,PF-E-SBRN Acum se tipăreste toată
                                        partea numărului dinainte de
                                        E.
                    LD    A,+45         Se introduce codul caracter
                                        pentru 'E'.
                    RST   0010,PRINT-A-1 Se tipăreste 'E'.
                    LD    C,D           Acum exponentul este în C
                                        pentru a fi tipărit.
                    LD    A,C           Si în A pentru a fi testat.
                    AND   A             I se testează semnul.
                    JP    P,2F83,FP-E-POS Salt dacă este pozitiv.
                    NEG                 Altfel, este negat în A.
                    LD    C,A           Apoi este copiat înapoi în C
                                        pentru a fi tipărit.
                    LD    A,+2D         Se introduce codul caracter
                                        pentru '-'.
                    JR    2F85,PF-E-SIGN Salt pentru a tipări semnul.
2F83 PF-E-POS       LD    A,+2B         Se introduce codul caracter
                                        pentru '+'.
2F85 PF-E-SIGN      RST   0010,PRINT-A-1 Acum se tipăreste semnul '+'
                                        sau '-'.
                    LD    B,+00         BC contine exponentul pentru
                                        a fi tipărit.
                    JP    1A1B,OUT-NUM  Salt înapoi pentru a- tipări
                                        si sfisit.
```

THE 'CA=10*A+C' SUBROUTINE (SUBRUTINA CA=10*A+C')

Această subrutină este apelată de subrutina PRINT-FP pentru a multiplica fiecare octet din D'E'DE cu 10 si returnează partea întreagă a rezultatului în registrul C. La intrare, registrul A contine octetul ce trebuie multiplicat cu 10 iar registrul C contine încă transportul octetului anterior. La revenire, registrul A contine octetul rezultat iar registrul C contine transportul mai departe al octetului următor.

```
2F8B CA=10*A+C   PUSH  DE          Salvarea oricărei perechi DE
                                   în utilizare.
                 LD    L,A         Se copiază înmultitorul din
                 LD    H,+00       A în HL.
                 LD    E,L         Se copiază si în DE.
                 LD    D,H
                 ADD   HL,HL       Se dublează HL.
                 ADD   HL,HL       Se dublează încă o dată.
                 ADD   HL,DE       Adunare DE pentru a obtine
                                   HL=5*A.
                 ADD   HL,HL       Se dublează din nou: acum
                                   HL=10*A.
                 LD    E,C         Se copiază C în DE (D este
                                   zero) pentru adunare.
                 ADD   HL,DE       Acum HL=10*A+C.
                 LD    C,H         H este copiat în C.
                 LD    A,L         L este copiat în A,completînd
                                   taskul.
                 POP   DE          Se readuce registrul pereche
                                   DE.
                 RET               Sfîrsit.
```

THE 'PREPARE TO ADD' SUBROUTINE (SUBRUTINA 'PREGATIRE PENTRU ADUNARE')
Această subrutină este prima din cele patru subrutine care sînt folosite de principalele rutine de operatii aritmetice - SUBTRACTION (scădere9, ADDITION (adunare), MULTIPLICATION (înmultire) si DIVISION (împărtire).

Această subrutină particulară pregăteste un număr în virgulă mobilă pentru adunare, mai ales prin înlocuirea bitului de semn cu un bit numeric adevărat 1, si negarea numărului (complementul fată de doi) dacă este negativ. Exponentul este returnat în registrul A si primul octet este setat pe 00 hex. pentru un număr pozitiv si pe FF hex. pentru un număr negativ.

```
2F9B PREP-ADD    LD    A,(HL)      Se transferă exponentul în A.
                 LD    (HL),+00    Se consideră un număr pozitiv
                 AND   A           Dacă numărul este zero,
                 RET   Z           atunci pregătirea s-a si
                                   terminat.
                 INC   HL          Acum se indică octetul de
                                   semn.
                 BIT   7,(HL)      Se setează fanionul zero dacă
                                   numărul este pozitiv.
                 SET   7,(HL)      Se reface bitul numeric
                                   adevărat.
                 DEC   HL          Se indică din nou primul
                                   octet.
                 RET   Z           Numerele pozitive au fost
                                   pregătite, dar numerele
                                   negative trebuie
                                   complementate fată de doi.
                 PUSH  BC          Salvarea oricărui exponent
                                   anterior.
                 LD    BC,+0005    Sînt 5 octeti care trebuie
                                   tratati.
                 ADD   HL,BC       Se indică unul după ultimul
                                   octet.
                 LD    B,C         Se transferă '5' în B.
                 LD    C,A         Se salvează exponentul în C.
                 SCF               Se setează fanionul zero
                                   pentru negare.
2FAF NEG-BYTE    DEC   HL          Se indică rînd pe rînd
                                   fiecare octet.
                 LD    A,(HL)      Se aduce fiecare octet.
                 CPL               Se face complementul fată de
                                   unu al octetului.
                 ADC   A,+00       Se adună transportul pentru
                                   negare.
                 LD    (HL),A      Se reface octetul.
                 DJNZ  2FAF,NEG-BYTE Se ciclează de '5' ori.
                 LD    A,C         Se reface exponentul în A.
                 POP   BC          Se reface orice exponent
                                   anterior.
                 RET               Sfîrsit.
```

THE 'FETCH TWO NUMBERS' SUBROUTINE (SUBRUTINA 'ADUCERE DOUA NUMERE')
Această subrutină este apelată de ADDITION, MULTIPLICATION si DIVISION pentru a aduce două numere din stiva calculatorului si pentru a le pune în registrii inclusiv în registrii de schimb.

La intrarea în subrutină registrul pereche HL indică primul octet al primului număr iar registrul pereche DE indică primul octet al celui de al

doilea număr.
Cînd subrutina este apelată din MULTIPLICATION sau DIVISION semnul rezultatului este salvat în al doilea octet al primului număr.

```
2FBA FETCH-TWO   PUSH  HL          Salvare HL.
                 PUSH  HL          Salvare AF.

Se apelează cei cinci octeti ai primului număr - M1, M2, M3, M4 & M5
si ai primului număr                           - N1, N2, N3, N4 & N5

                 LD    C,(HL)      M1 în C.
                 INC   HL          Următorul
                 LD    B,(HL)      M2 în B.
                 LD    (HL),A      Se copiază semnul
                                   rezultatului în (HL).
                 INC   HL          Următorul.
                 LD    A,C         M1 în A.
                 LD    C,(HL)      M3 în C.
                 PUSH  BC          Se salvează M2 & M3 în stiva
                                   masinii.
                 INC   HL          Următorul.
                 LD    C,(HL)      M4 în C.
                 INC   HL          Următorul.
                 LD    B,(HL)      M5 în B.
                 EX    DE,HL       Acum HL indică N1.
                 LD    D,A         M1 în D.
                 LD    E,(HL)      N1 în E.
                 PUSH  DE          Se salvează M1 & N1 în stiva
                                   masinii.
                 INC   HL          Următorul.
                 LD    D,(HL)      N2 în D.
                 INC   HL          Următorul.
                 LD    E,(HL)      N3 în E.
                 PUSH  DE          Se salvează N2 & N3 în stiva
                                   masinii.
                 EXX               Se aduc registrii de schimb.
                 POP   DE          N2 în D' & N3 în E'.
                 POP   HL          M1 în H' & N1 în L'.
                 POP   BC          M2 în B' & M3 în C'.
                 EXX               Se aduce setul original de
                                   registrii.
                 INC   HL          Următorul.
                 LD    D,(HL)      N4 în D.
                 INC   HL          Următorul.
                 LD    E,(HL)      N5 în E.
                 POP   AF          Se restaurează AF original.
                 POP   HL          Se restaurează HL original.
                 RET               Sfîrsit.
```

Sumar: M1 - M5 sînt în H',B',C',C,B.
N1 - N5 sînt în L',D',E',D,E.
HL indică primul octet al primului număr.

THE 'SHIFT ADDEND' SUBROUTINE (SUBRUTINA 'DEPLASARE TERMEN ADUNARE')
Această subrutină deplasează un număr în virgulă mobilă pînă la 32 zecimale, 20 hex, plasat la dreapta liniei asa cum trebuie pentru adunare. Numărul cu exponentul mai mic a fost pus în pozitia termenului adunării înaintea apelării subrutinei. Fiecare depăsire la dreapta, în transport, este adunată înapoi la număr. Dacă diferenta exponentilor este mai mare de 32 zecimal sau transportul revine la începutul numărului atunci numărul este setat pe zero astfel încît adunarea nu va modifica alte numere (termeni ai adunării).

```
2FDD SHIFT-FP    AND   A           Dacă diferenta exponentilor
                 RET   Z           este zero, subrutina revine
                 CP    +21         imediat. Dacă diferenta este
                 JR    NC,2FF9,ADDEND-0  mai mare decît 20 hex, salt
                                   înainte.
                 PUSH  BC          Se salvează BC pentru scurt
                                   timp.
                 LD    B,A         Se transferă diferenta
                                   exponent în B pentru a număra
                                   deplasările la dreapta.
2FE5 ONE-SHIFT   EXX               Deplasare aritmetică la
                 SRA   L           dreapta pentru L', păstrînd
                                   bitul marcator de semn.
                 RR    D           Rotire la dreapta cu
                 RR    E           transport D',E',D&E.
                 EXX               Astfel se deplasează toti cei
                 RR    D           cinci octeti ai numărului la
```

```
                RR    D                       dreapta de atîtea ori de cîte
                                              ori numără B.
                DJNZ  2FE5,ONE-SHIFT          Se ciclează pînă cînd B
                                              ajunge la zero.
                POP   BC                      Se reface BC original.
                RET   NC                      Revenire dacă nu se mai
                                              găseste nici un transport.
                CALL  3004,ADD-BACK           Regăsire transport.
                RET   NZ                      Se revine numai dacă
                                              transportul revine înapoi (în
                                              acest caz nu este nimic de
                                              adunat).
2FF9 ADDEND-0   EXX                           Se aduce L',D'&E'.
                XOR   A                       Se sterge registrul A.
2FFB ZEROS-4/5  LD    L,+00                   Se setează termenul adunării
                LD    D,A                     la zero în D',E',D&E,împreună
                LD    E,L                     cu octetul marcator
                EXX                           (indicator semn) L', care a
                LD    DE,+0000                fost 00 hex pentru un număr
                                              pozitiv si FF hex pentru un
                                              număr negativ. ZEROS-4/5
                                              produce numai 4 octeti zero
                                              cînd apelează în apropiere
                                              la 3160.
                RET                           Sfîrsit.
```

THE 'ADD-BACK' SUBROUTINE (SUBRUTINA 'ADD-BACK')
Această subrutină adună înapoi în număr orice transport care a depăsit la dreapta. In caz extrem, transportul revine înapoi la stînga numărului.
Cînd această subrutină este apelată în timpul adunării, această fluctuatie înseamnă că o mantisă a lui 0,5 a fost deplasată cu totul 32 locuri la dreapta, si termenul adunării va fi acum setat pe zero; cînd este apelată din MULTIPLICATION, înseamnă că exponentul trebuie incrementat, iar aceasta poate rezulta din depăsire.

```
3004 ADD-BACK   INC   E                       Se adună transportul la
                                              octetul cel mai din dreapta.
                RET   NZ                      Se revine dacă nu este
                                              depăsire la stînga.
                INC   D                       Se continuă cu următorul
                                              octet.
                RET   NZ                      Revenire dacă nu este
                                              depăsire la stînga.
                EXX                           Se aduce următorul octet.
                INC   E                       Si acesta se incrementează.
                JR    NZ,300D,ALL-ADDED       Salt dacă nu este depăsire.
                INC   D                       Se incrementează ultimul
                                              octet.
300D ALL-ADDED  EXX                           Se readuc registrii originari
                RET                           Sfîrsit.
```

THE 'SUBTRACTION' OPERATION (OPERATIA 'SUBTRACTION' (scădere))
(Deplasament 03 - vezi CALCULATE după: 'subtract')

Această subrutină doar schimbă semnul scăzătorului si îl transportă în ADDITION.
De notat că HL indică descăzutul si DE indică scăzătorul. (Vezi ADDITION pentru mai multe detalii).

```
300F SUBTRACT   EX    DE,HL                   Se interschimbă indicatorii.
                CALL  346E,NEGATE             Se schimbă semnul
                                              scăzătorului.
                EX    DE,HL                   Se interschimbă indicatorii
                                              înapoi si se continuă în
                                              ADDITION.
```

THE 'ADDITION' OPERATION (OPERATIA 'ADDITION' (adunare))
(Deplasament 0F - vezi CALCULATE după: 'addition')

Prima din cele mai importante trei subrutine aritmetice, această subrutină execută adunarea în virgulă mobilă a două numere, fiecare cu 4 octeti pentru mantisă si 1 octet pentru exponent. In aceste trei subrutine, cele două numere din vîrful stivei calculatorului sînt adunate/înmultite/împărtite pentru a obtine un număr în vîrful stivei calculatorului, o 'ultima valoare'. HL indică al doilea număr din vîrf, primul termen al adunării/deînmultitul/deîmpărtitul. DE indică numărul din vîrful stivei calculatorului, al doilea termen al adunării/înmultitorul/împărtitorul. Apoi HL indică rezultanta 'ultima valoare' a cărei adresa se poate considera ca fiind în STKEND-5.
Dar subrutina de adunare testează mai întîi dacă cele două numere ce

trebuie adunate sînt 'numere întregi mici'. Dacă sînt, le adună simplu în HL si BC, si pune rezultatul direct în stivă. Nu este necesară complementarea fată de doi înainte sau după adunare, deoarece asemenea numere sînt continute în stivă în forma complementată fată de doi, gata pentru adunare.

```
3014 addition   LD   A,(DE)          Se testează dacă primii
                OR   (HL)            octeti ai ambelor numere
                                     sînt zero.
                JR   NZ,303E,FULL-ADDN  Dacă nu, salt pentru adunare
                                     deplină.
                PUSH DE              Se adună indicatorul la al
                                     doilea număr.
                INC  HL              Se indică al doilea octet al
                PUSH HL              primului număr si se salvează
                                     indicatorul.
                INC  HL              Se indică cel mai putin
                                     semnificativ octet.
                LD   E,(HL)          acesta este adus în E.
                INC  HL              Se indică octetul cel mai
                                     semnificativ.
                LD   D,(HL)          Acesta este adus în D.
                INC  HL              Se trece la al doilea octet
                INC  HL              al celui de al doilea număr.
                INC  HL
                LD   A,(HL)          Acesta este adus în A (acesta
                                     este octetul semn).
                INC  HL              Se indică octetul cel mai
                                     putin semnificativ.
                LD   C,(HL)          Acesta este adus în C.
                INC  HL              Se indică octetul cel mai
                                     semnificativ.
                LD   B,(HL)          Aceasta este adus în B.
                POP  HL              Se aduce indicatorul
                EX   DE,HL           octetului de semn al primului
                                     număr; este pus în DE, iar
                                     numărul în HL.
                ADD  HL,BC           Se execută adunarea:
                                     rezultatul în HL.
                EX   DE,HL           Rezultatul în DE, octetul de
                                     semn în HL.
                ADD  A,(HL)          Se adună octetul de semn si
                RRCA                 transportul în A; aceasta va
                                     detecta dacă este depăsire.
                ADC  A,+00           Un A diferit de zero indică
                                     depăsire.
                JR   NZ,303C,ADDN-OFLW  Salt pentru a reseta
                                     indicatorii si pentru adunare
                                     deplină.
                SBC  A,A             Definire octet de semn corect
                                     pentru rezultat.
3032            LD   (HL),A          Se stochează în stivă,
                INC  HL              se indică următoarea locatie.
                LD   (HL),E          Se stochează octetul cel mai
                                     putin semnificativ al
                                     rezultatului.
                INC  HL              Se indică următoarea locatie.
                LD   (HL),D          Se stochează octetul cel mai
                                     semnificativ al rezultatului.
                DEC  HL              Se mută indicatorul înapoi la
                DEC  HL              adresa primului octet al
                DEC  HL              rezultatului.
                POP  DE              Se readuce STKEND în DE.
                RET                  Sfîrsit.
```

De notat că numărul -65536 zecimal poate apărea aici în forma 00 FF 00 00 00 ca rezultatul adunării a două numere întregi mici negative, de exemplu -65000 si -536. Este mai simplu de stocat în această formă. Aceasta este o greseală. Sistemul Spectrum nu poate trata acest număr.
Mai multe functii îl tratează ca un zero, si este tipărit ca -1E38 obtinut prin tratarea ca 'minus zero' într-un format ilegal.
Un remediu posibil ar fi testarea acestui număr în jurul octetului 3032 si, dacă este prezent, să se facă al doilea octet 80 hex si primul octet 91 hex, astfel umplînd cei cinci octeti ai numărului în virgulă mobilă, adică 91 80 00 00 00, care ridică nici o problemă. Vezi si precizările din 'truncate' de după, înaintea octetului 3225, si Anexa.

```
303C ADDN-OFLW  DEC  HL              Se readuce indicatorul la
                                     primul număr.
                POP  DE              Se readuce indicatorul al al
```

```
303E FULL-ADDN      CALL  3293,RE-ST-TWO      doilea număr.
                                              Se restochează ambele numere
                                              umplînd cei cinci octeti ai
                                              formei în virgulă mobilă.
```

Subrutina ADDITION în totalitatea ei apelează mai întîi PREP-ADD pentru fiecare număr, apoi aduce cele două numere din stiva calculatorului si-l pune pe cel cu exponentul mai mic în pozitia de prim termen al adunării. Apoi apelează SHIFT-FP pentru a deplasa acest termen al adunării cu 32 zecimal locuri la dreapta pentru a-l alinia pentru adunare. Adunarea actuală se face în cîtiva octeti, se face o singură deplasare pentru transport (depăsire la stînga) dacă este necesar, rezultatul este complementat fată de doi dacă este negativ, si orice depăsire aritmetică este raportată; altfel subrutina sare la TEST-NORM pentru a normaliza rezultatul si îl pune în stivă cu bitul se semn corect introdus în al doilea octet.

```
                    EXX                       Se interschimbă registrii.
                    PUSH  HL                  Salvarea următoarei adrese
                                              corecte.
                    EXX                       Se interschimbă registrii.
                    PUSH  DE                  Se salvează indicatorul
                                              primului termen al adunării.
                    PUSH  HL                  Se salvează indicatorul
                                              celui de al doilea termen al
                                              adunării.
                    CALL  2F9B,PREP-ADD       Se pregăteste al doilea
                                              termen al adunării.
                    LD    B,A                 Se salvează exponentul lui în
                                              B.
                    EX    DE,HL               Se interschimbă indicatorii.
                    CALL  2F9B,PREP-ADD       Se pregăteste primul termen
                                              al adunării.
                    LD    C,A                 Se salevază exponentul lui.
                    CP    B                   Dacă primul exponent este mai
                    JR    NC,3055,SHIFT-LEN   mic, se păstrează primul
                    LD    A,B                 număr în pozitia primului
                    LD    B,C                 termen al adunării; altfel se
                    EX    DE,HL               schimbă din nou exponentii si
                                              indicatorii.
3055 SHIFT-LEN      PUSH  AF                  Se salvează exponentul mai
                                              mare în A.
                    SUB   B                   Diferenta dintre exponenti
                                              este lungimea deplasării la
                                              dreapta.
                    CALL  2FBA,FETCH-TWO      Se aduc cele două numere din
                                              stivă.
                    CALL  2FDD,SHIFT-FP       Se deplasează la dreapta
                                              primul termen al adunării.
                    POP   AF                  Se readuce exponentul mai
                                              mare.
                    POP   HL                  HL va indica rezultatul.
                    LD    (HL),A              Se stochează exponentul
                                              rezultatului.
                    PUSH  HL                  Se salvează indicatorul din
                                              nou.
                    LD    L,B                 M4 în H & M5 în L,
                    LD    H,C                 (vezi FETCH-TWO).
                    ADD   HL,DE               Se adună cei doi octeti din
                                              dreapta.
                    EXX                       N2 în H' & N3 în L',
                    EX    DE,HL               (vezi FETCH-TWO).
                    ADC   HL,BC               Se adună octetii din stînga
                                              cu transportul.
                    EX    DE,HL               Rezultatul înapoi în D'E'.
                    LD    A,H                 Se adună H',L' si transportul
                    ADC   A,L                 mecanismul rezultat asigurînd
                    LD    L,A                 că se va apela o singură
                    RRA                       deplasare la dreapta dacă
                    XOR   L                   suma a 2 numere pozitive a
                    EXX                       dat depăsire la stînga, sau
                                              suma a două numere negative
                                              nu a dat depăsire la stînga.
                    EX    DE,HL               Acum rezultatul este în
                                              DED'E'.
                    POP   HL                  Se aduce indicatorul la
                                              exponent.
                    RRA                       Test pentru deplasare (H',L'
                    JR    NC,307C,TEST-NEG    au fost 00 hex pentru numere
                                              pozitive si FF hex pentru
```

```
                    LD    A,+01          numere negative).
                                         A numără o singură deplasare
                                         la dreapta.
                    CALL  2FDD,SHIDT-FP  Apelare deplasare.
                    INC   (HL)           Se adună 1 la exponent;
                    JR    Z,309F,AD-REP-6 aceasta poate conduce la o
                                         depăsire aritmetică.
307C TEST-NEG       EXX                  Test pentru rezultat negativ;
                    LD    A,L            se aduce bitul semn al lui L'
                    AND   +80            în A (acesta indică acum
                    EXX                  corect semnul rezultatului).
                    INC   HL             Acesta se va stoca în al
                    LD    (HL),A         doilea octet poziție a
                    DEC   HL             rezultatului în stiva
                                         calculatorului.
                    JR    Z,3DA5,GO-NC-MLT Dacă este zero, atunci nu se
                                         complementează față de doi
                                         rezultatul.
                    LD    A,E            Se aduce primul octet.
                    NEG                  Se face negarea lui.
                    CCF                  Se complementează transportul
                    LD    E,A            pentru a continua negarea si
                                         se stochează octetul.
                    LD    A,D            Se aduce următorul octet.
                    CPL                  Se complementează față de unu
                    ADC   A,+00          Adunare în transport pentru
                                         negare.
                    LD    D,A            Se stochează octetul.
                    EXX                  Se trece la aducerea
                    LD    A,E            următorului octet în
                                         registrul A.
                    CPL                  Se complementează față de unu
                    ADC   A,+00          Adunare în transport pentru
                                         negare.
                    LD    E,A            Se stochează octetul.
                    LD    A,D            Se aduce ultimul octet.
                    CPL                  Se complementează față de unu
                    ADC   A,+00          Se adună în transport pentru
                                         negare.
                    JR    NC,3DA3,END-COMPL Se execută dacă nu este
                                         transport.
                    RRA                  Altfel, se duce .5 în mantisă
                    EXX                  si se adună 1 la exponent;
                    INC   (HL)           acesta este necesar cînd se
                                         adună două numere negative
                                         pentru a da exact puterea lui
                                         2, si poate conduce la o
                                         depăsire aritmetică.
309F ADD-REP-6      JP    Z,31AD,REPORT-6 Se prezintă eroare dacă se
                                         cere.
                    EXX
30A3 END-COMPL      LD    D,A            Se stochează ultimul octet.
                    EXX
30A5 GO-NC-MLT      XOR   A              Se șterge fanionul de
                                         transport.
                    JP    3155,TEST-NORM Ieșire prin TEST-NORM.
```

THE 'HL=HL*DE' SUBROUTINE (SUBRUTINA 'HL=HL*DE')

Această subrutină este apelată de 'GET-HL*DE' si de 'MULTIPLICATION' pentru a efectua înmultirea pe 16 biti ca atare.

Orice depăsire a celor 16 biti admisi este tratată cu o revenire din subrutină.

```
30A9 HL=HL*DE       PUSH  BC             Este salvat BC.
                    LD    B,+10          Este o înmultire pe 16 biti.
                    LD    A,H            A contine octetul cel mai
                                         semnificativ.
                    LD    C,L            C contine octetul cel mai
                                         putin semnificativ.
                    LD    HL,+0000       Se initializează rezultatul
                                         pe zero.
30B1 HL-LOOP        ADD   HL,HL          Se dublează rezultatul.
                    JR    C,30BE,HL-END  Salt dacă este depăsire.
                    RL    C              Se roteste bitul 7 din C în
                                         transport.
                    RLA                  Se roteste bitul de transport
                                         în bitul 0 si bitul 7 în
                                         fanionul de transport.
                    JR    NC,30BC,HL-AGAIN Salt dacă fanionul de
```

```
                        ADD   HL,DE          transport este resetat.
                                             Altfel se adună DE imediat.
                        JR    C,30BE,HL-END  Salt dacă este depăşire.
30BC HL-AGAIN           DJNZ  30B1,HL-LOOP   Se ciclează de 16 ori.
30BE HL-END             POP   BC             Se readuce BC.
                        RET                  Sfîrşit.
```

THE 'PREPARATE TO MULTIPLY OR DIVIDE' SUBROUTINE (SUBRUTINA 'PREGATIRE PENTRU INMULTIRE SAU IMPARTIRE')
Această subrutină pregăteşte un număr în virgulă mobilă pentru înmultire sau împărtire, revenind cu transportul setat dăcă numărul este zero, aducînd semnul rezultatului în registrul A, si înlocuind bitul semn în număr prin bitul numeric adevărat, 1.

```
30C0 PREP-M/D           CALL  34E9,TEST-ZERO Dacă numărul este zero, se
                        RET   C              revine cu fanionul de
                                             transport resetat.
                        INC   HL             Se indică octetul de semn.
                        XOR   (HL)           Se aduce semnul rezultatului
                                             în A (semn dorit dă plus,
                                             semn nedorit dă minus); de
                                             asemenea se resetează
                                             fanionul de transport.
                        SET   7,(HL)         Se setează bitul numeric
                                             adevărat.
                        DEC   HL             Se indică din nou următorul
                                             exponent.
                        RET                  Se revine cu fanionul de
                                             transport resetat.
```

THE 'MULTIPLICATION' OPERATION (OPERATIA 'MULTIPLICATION' (înmultire))
(Deplasament 04 - vezi CALCULATE după: 'multiply')

Această subrutină testează mai întîi dacă cele două numere care trebuie înmultite sînt 'întregi mici'. Dacă sînt, subrutina foloseste INT-FETCH pentru a aduce numerele din stivă, apoi HL=HL*DE pentru a le înmulti si INT-STORE pentru a returna rezultatul în stivă. orice depăşire a acestei 'înmultiri scurte' (adică dacă rezultatul nu este el însusi un 'întreg mic') cauzează un salt la înmultirea celor cinci octeti ai formei în virgulă mobilă (vezi după).

```
30CA multiply           LD    A,(DE)         Se testează care din primii
                        OR    (HL)           octeti ai ambelor numere este
                                             zero.
                        JR    NZ,30F0,MULT-LONG Dacă nu este zero, salt
                                             pentru înmultire 'lungă'.
                        PUSH  DE             Salvare indicatori: pentru
                                             al doilea număr.
                        PUSH  HL             Si pentru primul număr.
                        PUSH  DE             Si din nou pentru al doilea
                                             număr.
                        CALL  2D7F,INT-FETCH Se aduce semnul în C, numărul
                                             în DE.
                        EX    DE,HL          Acum numărul în HL.
                        EX    (SP),HL        Numărul în stivă, al doilea
                                             indicator în HL.
                        LD    B,C            Se salvează primul semn în B.
                        CALL  2D7F,INT-FETCH Se aduce al doilea semn în C,
                                             numărul în DE.
                        LD    A,B            Se formează semnul
                        XOR   C              rezultatului în A: semn dorit
                                             dă plus (00, semn nedorit dă
                                             minus (FF).
                        LD    C,A            Se stochează semnul
                                             rezultatului în C.
                        POP   HL             Se readuce primul număr în HL
                        CALL  30A9,HL=HL*DE  Se efectuează înmultirea
                                             actuală.
                        EX    DE,HL          Se stochează rezultatul în DE
                        POP   HL             Se readuce indicatorul
                                             primului număr.
                        JR    C,30EF,MULT-OFLW Salt în depăşirea înmultirii
                                             'depline'.
30E5                    LD    A,D            Acesti 5 octeti asigură că
                        OR    E              00 FF 00 00 00 este înlocuit
                        JR    NZ,30EA,MULT-RSLT cu zero; ei nu ar fi necesari
                        LD    C,A            dacă acest număr ar fi fost
                                             exclus din sistem (vezi după
                                             303B deasupra).
30EA MULT-RSLT          CALL  2D8E,INT-STORE Acum se stochează rezultatul
```

```
                        POP   DE          în stivă.
                        RET               Se readuce STKEND în DE.
                                          Sfîrsit.
30EF MULT-OFLW          POP   DE          Se readuce indicatorul celui
                                          de al doilea număr.
30F0 MULT-LONG          CALL  3293,RE-ST-TWO  Se restochează ambele numere
                                          cu toti cei cinci octeti ai
                                          formei în virgulă mobilă.

                        XOR   A           A este setat cu 00 hex,astfel
                                          încît semnul primului număr
                                          trece în A.
                        CALL  30C0,PREP-M/D  Se pregăteste primul număr,
                        RET   C           se revine dacă este zero.
                                          (Rezultatul este deja zero.)
                        EXX               Se interchimbă registrii.
                        PUSH  HL          Salvarea următoarei adrese
                                          corecte.
                        EXX               Se interschimbă registrii.
                        PUSH  DE          Se salvează indicatorul
                                          înmultitorului.
                        EX    DE,HL       Se interschimbă indicatorii.
                        CALL  30C0,PREP-M/D  Se pregăteste al doilea
                                          număr.
                        EX    DE,HL       Se interschimbă indicatorii
                                          din nou.
                        JR    C,315D,ZERO-RSLT  Salt înainte dacă al doilea
                                          număr este zero.
                        PUSH  HL          Se salvează indicatorul
                                          rezultatului.
                        CALL  2FBA,FETCH-TWO  Se aduc cele două numere din
                                          stivă.
                        LD    A,B         MS în A (vezi (FETCH-TWO).
                        AND   A           Pregătire pentru scădere.
                        SBC   HL,HL       Se intializează HL cu zero
                                          pentru rezultat.
                        EXX               Se interschimbă registrii.
                        PUSH  HL          Se salvează M1 & N1 (vezi
                                          FETCH-TWO).
                        SBC   HL,HL       De asemenea se initializează
                                          H'L' pentru rezultat.
                        EXX               Se interschimbă registrii.
                        LD    B,+21       B numără 33 zecimal, 21 hex,
                                          rotiri.
                        JR    3125,STRT-MLT  Salt înainte în buclă.

- Acum se introduce bucla de înmultire.

3114 MLT-LOOP           JR    NC,311B,NO-ADD  Salt înainte la NO-ADD dacă
                                          nu este transport (adică dacă
                                          bitul deînmultitului a fost
                                          resetat).
                        ADD   HL,DE       Altfel, se adună înmutitorul
                        EXX               in D'E'DE (vezi FETCH-ADD) în
                        ADC   HL,DE       rezultatul care a fost
                        EXX               construit în H'L'HL.
311B NO-ADD             EXX               Dacă înmultitorul a fost
                        RR    H           sau nu adunat, se deplasează
                        RR    L           rezultatul la dreapta în
                        EXX               H'L'HL, adică deplasarea se
                        RR    H           face rotind fiecare octet cu
                        RR    L           transportul, astfel încît
                                          orice bit care trece în
                                          transport este scos de
                                          următorul octet, si
                                          deplasarea continuă în
                                          B'C'BA.
3125 STRT-MLT           EXX               Se deplasează la dreapta
                        RR    B           înmultitorul în B'C'CA (vezi
                        RR    C           FETCH-TWO & deasupra).
                        EXX
                        RR    C           Un bit final coborît în
                        RRA               transport va atrage o altă
                                          adunare a înmultitorului la
                                          rezultat.
                        DJNZ  3114,MLT-LOOP  Se ciclează de 33 de ori
                                          pentru a aduce toti bitii.
                        EX    DE,HL       Se trece rezultatul din:
                        EXX
```

```
                EX      DE,HL           H'L'HL în D'E'DE.
                EXX
```

Acum se adună exponentii împreună.

```
                POP     BC              Se readuc exponentii - M1&N1.
                POP     HL              Se readuce indicatorul
                                        octetului exponent.
                LD      A,B             Se aduce suma celor doi
                ADD     A,C             octeti exponenti în A, si
                                        transportul corect.
                JR      NZ,313B,MAKE-EXPT  Dacă suma este egală cu zero,
                AND     A               atunci se sterge transportul;
                                        altfel se lasă neschimbat.
313B MAKE-EXPT  DEC     A               Pregătire pentru incrementare
                CCF                     a exponentului cu 80 hex.
```

Restul subrutinei este comun si pentru MULTIPLICATION si pentru DIVISION.

```
313D DIVN-EXPT  RLA                     Acesti cîtiva octeti foarte
                CCF                     iscusiti corectează octetul
                                        exponent.
                RRA                     Rotind la stînga apoi la
                                        dreapta se aduce octetul
                                        exponent (adevăratul exponent
                                        plus 80 hex) în A.
                JP      P,3146,OFLW1-CLR  Dacă fanionul de semn este
                                        resetat, nu este necesară
                                        prezentarea unei depăsiri
                                        aritmetice.
                JR      NC,31AD,REPORT-6  Se raportează depăsire dacă
                                        transportul este resetat.
                AND     A               Acum se sterge transportul.
3146 OFLW1-CLR  INC     A               Octetul exponent este acum
                JR      NZ,3151,OFLW2-CLR  complet;Dar dacă A este zero,
                JR      C,3151,OFLW2-CLR   atunci este necesară o
                                        verificare a depăsirii mai
                                        departe.
                EXX                     Dacă transportul nu este
                BIT     7,D             setat si rezultatul este deja
                EXX                     în forma normală (bitul 7 al
                JR      NZ,31AD,REPORT-6  lui D' setat) atunci trebuie
                                        raportată depăsire; dar dacă
                                        bitul 7 al lui D' este
                                        resetat, rezultatul este
                                        chiar în interval, adică
                                        chiar sub 2**127.
3151 OFLW2-CLR  LD      (HL),A          Se stochează octetul exponent
                EXX                     Se trece al cincilea octet
                LD      A,B             rezultat în A pentru
                EXX                     secventa de normalizare,
                                        adică depăsirea din L B'.
```

Restul subrutinei lucrează cu normalizarea si este comună pentru toate rutinele aritmetice.

```
3155 TEST-NORM  JR      NC,316C,NORMALISE  Dacă nu este transport atunci
                                        se normalizează acum.
                LD      A,(HL)          Altfel, se lucrează cu sub
                AND     A               nivel (rezultat zero) sau în
3159 NEAR-ZERO  LD      A,+80           jur de sub nivel (rezultat
                                        2**-128);
                JR      Z,315E,SKIP-ZERO  exponentul este returnat în
315D ZERO-RSLT  XOR     A               A, se testează dacă A este
315E SKIP-ZERO  EXX                     zero (cazul2**-128) si dacă
                AND     D               este asa se execută 2**-128
                                        dacă numărul este normal,
                                        altfel este produs zero.
                CALL    2FFB,ZEROS-4/5  Apoi exponentul trebuie setat
                RLCA                    pe zero (pentru zero) sau 1
                                        (pentru 2**-128).
                LD      (HL),A          Se reface octetul exponent.
                JR      C,3195,OFLOW-CLR  Salt dacă este cazul 2**-128.
                INC     HL              Altfel, se pune zero în al
                LD      (HL),A          doilea octet al rezultatului
                DEC     HL              în stiva calculatorului.
                JR      3195,OFLOW-CLR  Salt înainte pentru a
                                        transfera rezultatul.
```

Normalizarea operatiei actuale.

```
316C NORMALISE    LD    B,+20          Se normalizează rezultatul
316E SHIFT-ONE    EXX                  făcînd 32 zecimal, 20 hex,
                  BIT   7,D            deplasări la stînga în D'E'DE
                  EXX                  (cu A alăturat) pînă cînd
                  JR    NZ,3186,NORML-NOW  bitul 7 din D' este setat. A
                  RLCA                 contine zero după adunare asa
                  RL    E              că nu se cîstigă si nu se
                  RL    D              pierde nici o precizie; A
                  EXX                  contine al cincilea octet din
                  RL    E              B' după înmultire sau
                  RL    D              împărtire; dar cum numai în
                  EXX                  jur de 32 de biti pot fi
                                       corectati, nu se pierde din
                                       precizie; De notat că A este
                                       rotit circular, cu ramură în
                                       transport ... eventual un
                                       proces întîmplător.
                  DEC   (HL)           Exponentul este decrementat
                                       la fiecare deplasare.
                  JR    Z,3159,NEAR-ZERO  Dacă exponentul devine zero,
                                       atunci numărul din 2**-129
                                       este rotunjit la 2**-128.
                  DJNZ  316E,SHIFT-ONE  Se ciclează înapoi de 32 de
                                       ori.
                  JR    315D,ZERO-RSLT  Dacă bitul 7 nu devine 1
                                       niciodată, atunci întregul
                                       rezultat trebuie să fie zero.
```

Se termină normalizarea luînd în considerare 'transportul'.

```
3186 NORML-NOW    RLA                  După normalizare se adună
                  JR    NC,3195,OFLW-CLR  înapoi orice transport final
                                       care a trecut în A.
                  CALL  3004,ADD-BACK  Salt înainte dacă transportul
                  JR    NZ,3195,OFLW-CLR  nu oscilează direct înapoi.
                  EXX                  Dacă el poate oscila direct
                  LD    D,+80          înapoi atunci mantisa este
                  EXX                  setată pe 0.5 si exponentul
                                       este incrementat.
                  INC   (HL)           Această operatie poate
                  JR    Z,31AD,REPORT-6  conduce la o depăsire
                                       aritmetică (caz final).
```

Partea finală a subrutinei implică trecerea rezultatului în octetii rezervati pentru el în stiva calculatorului si resetarea indicatorilor.

```
3195 OFLOW-CLR    PUSH  HL            Se salvează indicatorul
                                      rezultatului.
                  INC   HL            Se indică octetul semn al
                                      rezultatului.
                  EXX                 Rezultatul este mutat din
                  PUSH  DE            registrii actuali, D'E'DE, în
                  EXX                 BCDE; si apoi în ACDE.
                  POP   BC
                  LD    A,B           Bitul de semn este readus din
                  RLA                 stiva sa temporară si
                  RL    (HL)          transferat în pozitia lui
                  RRA                 corectă a bitului 7 al
                                      primului octet al mantisei.
                  LD    (HL),A        Este stocat primul octet.
                  INC   HL            Următorul.
                  LD    (HL),C        Este stocat al doilea octet.
                  INC   HL            Următorul.
                  LD    (HL),D        Este stocat al treilea octet.
                  INC   HL            Următorul.
                  LD    (HL),E        Este stocat al patrulea octet
                  POP   HL            Se readuce indicatorul
                                      rezultatului.
                  POP   DE            Se readuce indicatorul celui
                                      de al doilea număr.
                  EXX                 Se interschimbă registrii.
                  POP   HL            Readucerea următoarei adrese
                                      corecte.
                  EXX                 Se interschimbă registrii.
                  RET                 Sfîrsit.
```

Prezentarea 6 - Depăsire aritmetică

```
31AD REPORT-6     RST   0008,ERROR-1        Se apelează rutina de tratare
                  DEFB  +05                 eroare.
```

THE 'DIVISION' OPERATION (OPERAȚIA 'DIVISION' (împărțire))
(Deplasament 05 - vezi CALCULATE după: 'division')

Această subrutină pregătește mai întîi împărțitorul apelînd PREP-M/D, prezentînd depășire aritmetică dacă este zero; apoi ea pregătește deîmpărțitul apelînd din nou PREP-M/D, revenind dacă este zero. Apoi se aduc cele două numere din stiva calculatorului si se împart mantisele lor în înțelesul unei uzuale redări a împărțirii, încercînd scăderea împărțitorului din deîmpărțit si redarea dacă este transport, altfel se adaugă 1 la cît. Precizia maximă se obține pentru împărțirea a 4 octeți, si după scăderea exponenților subrutina iese alăturînd ultima parte de la MULTIPLICATION.

```
31AF division     CALL  3293,RE-ST-TWO      Se utilizează forma lungă în
                                            virgulă mobilă.
                  EX    DE,HL               Se schimbă indicatorii.
                  XOR   A                   A este setat pe 00 hex, asa
                                            că semnul primului număr va
                                            merge în A.
                  CALL  30C0,PREP-M/D       Se pregăteste împărtitorul
                  JR    C,31AD,REPORT-6     si dacă este zero se
                                            raportează depășire
                                            aritmetică.
                  EX    DE,HL               Se schimbă indicatorii.
                  CALL  30C0,PREP-M/D       Se pregăteste deîmpărtitul si
                  RET   C                   se revine dacă este zero
                                            (rezultatul este deja zero).
                  EXX                       Se schimbă indicatorii.
                  PUSH  HL                  Se salvează următoarea adresă
                                            corectă.
                  EXX                       Se schimbă indicatorii.
                  PUSH  DE                  Salvare indicator împărtitor.
                  PUSH  HL                  Salvare indicator deîmpărtit.
                  CALL  2FBA,FETCH-TWO      Se adu cele două numere din
                                            stivă.
                  EXX                       Schimbare indicatori.
                  PUSH  HL                  Se salvează M1 & N1 în stiva
                                            masinii.
                  LD    H,B                 Se copiază cei patru octeti
                  LD    L,C                 ai deîmpărtitului din
                  EXX                       registrii B'C'CB (adică M2,
                  LD    H,C                 M3, M4 & M5 - vezi FETCH-TWO)
                                            în registrii H'L'HL.
                  LD    L,B
                  XOR   A                   Se sterge A si se resetează
                                            fanionul de transport.
                  LD    B,+DF               B va număra în sus de la -33
                                            la -1, în complement față de
                                            doi. DF la FF hex, ciclînd în
                                            minus si va sări din nou la
                                            zero pentru precizie mărită.
                  JR    31E2,DIV-START      Salt înainte în bucla de
                                            împărtire pentru prima
                                            încercare de scădere.
```

Acum se intră în bucla de împărtire.

```
31D2 DIV-LOOP     RLA                       Se deplasează rezultatul la
                  RL    C                   stînga în B'C'CA, deplasînd
                  EXX                       afară bitii ce sînt deja
                  RL    C                   acolo, scotînd 1 din
                  RL    B                   transport de cîte ori este
                  EXX                       setat, si rotind la stînga
                                            fiecare octet cu transportul
                                            pentru a ajunge la deplasarea
                                            bitului 32.
31DB DIV-34TH     ADD   HL,HL               Se transferă ce a mai rămas
                  EXX                       din deîmpărtit la stînga în
                  ADC   HL,HL               H'L'HL înainte de următoarea
                  EXX                       încercare de scădere; dacă un
                                            bit coboară în transport, se
                                            forțează neînapoierea si un
                                            bit pentru cît, care
                                            regăsește bitul pierdut si
                                            permite un împărtitor plin de
                  JR    C,31F2,SUBN-ONLY    32 biti.
```

```
31E2 DIV-START   SBC   HL,DE          Se încearcă scăderea
                 EXX                  împărtitorului din D'E'DE din
                 SBC   HL,DE          restul deîmpărtitului din
                 EXX                  H'L'HL; nu este nici un
                                      transport initial (vezi pasul
                                      anterior).
                 JR    NC,31F9,NO-RSTORE  Salt înainte dacă nu este
                                      transport.
                 ADD   HL,DE          Altfel se readuce, adică
                 EXX                  se adună înapoi împărtitorul.
                 ADC   HL,DE          Apoi se sterge transportul
                 EXX                  astfel încît acolo nu va fi
                 AND   A              nici un bit pentru cît
                                      (divizorul 'nu a mers').
                 JR    31FA,COUNT-ONE Salt înainte la numărător.
31F2 SUBN-ONE    AND   A              Se scade doar, fără nici o
                 SBC   HL,DE          înapoiere si se trece la
                 EXX                  setarea fanionului de
                 SBC   HL,DE          transport deoarece bitul
                 EXX                  pierdut al deîmpărtitului
                                      trebuie regăsit si folosit
                                      pentru cît.
31F9 NO-STORE    SCF                  Unu pentru cît în B'C'CA.
31FA COUNR-ONE   INC   B              Se trece prin buclă
                                      contorizînd unu.
                 JP    M,31D2,DIV-LOOP  Se ciclează de 32 de ori
                                      pentru toti bitii.
                 PUSH  AF             Se salvează fiecare bit 33
                                      pentru o precizie
                                      suplimentară (transportul
                                      prezent).
                 JR    Z,31E2,DIV-START  Se încearcă încă o dată
                                      scăderea pentru fiecare al
                                      34-lea bit; PUSH AF de
                                      deasupra salvează si acest
                                      bit.
```

Notă: Acest salt se face într-un loc greșit. Nici un al 34-lea bit nu va fi niciodată obținut fără a fi mai întîi deplasat la stînga. Apoi rezultatele importante ca 1/10 si 1/1000 nu sînt rotunjite așa cum ar trebui. Rotunjirea nu intervine niciodată cînd depinde de al 34-lea bit. Saltul trebuia să fi fost la 31DB DIV-34TH după: adică octetul 3200 hex în ROM trebuie să citească DA hex (218 zecimal) în locul lui E1 hex (225 zecimal).

```
                 LD    E,A            Acum se mută cei patru octeti
                 LD    D,C            care formează octetii
                 EXX                  mantisei rezultatului din
                                      B'C'CA în D'E'DE.
                 LD    E,C
                 LD    D,B
                 POP   AF             Apoi se pun bitii 34 si 33 în
                 RR    B              B' pentru a fi scosi dacă nu
                 POP   AF             este normalizare.
                 RR    B
                 EXX
                 POP   BC             Se readuc octetii exponent
                                      M1 & N1.
                 POP   HL             Readucere indicator rezultat.
                 LD    A,B            Se aduce diferența dintre cei
                 SUB   C              doi octeti exponent în A si
                                      se setează fanionul de
                                      transport dacă se cere.
                 JP    313D,DIVN-EXPT Iesire prin DIVN-EXPT.
```

THE 'INTEGER TRUNCATION TOWARDS ZERO' SUBROUTINE (SUBRUTINA 'TRUNCHIERE INTREGEATA DE ZERO')
(Deplasament 3A - vezi CALCULATE după: 'truncate')

Această subrutină (numită I(x)) redă rezultatul trunchierii numărului întreg x, 'ultima valoare', față de zero. Astfel, I(2,4) este 2 si I(-2,4) este -2. Subrutina revine imediat dacă x este în forma de 'întreg scurt' ('mic'). Ea redă zero dacă octetul exponent a lui x este mai mic decît 81 hex (ABS x este mai mic decît 1). Dacă I(x) este 'un întreg scurt' subrutina îl readuce în acea formă. x este redat dacă octetul exponent este A0 hex sau mai mare (x are partea neîntreagă nesemnificativă). Altfel numărul corect de octeti ai lui x sînt setati pe zero si, dacă este necesar, încă un octet este despărtit cu o mască.

```
3214 truncate    LD    A,(HL)         Se aduce octetul exponent al
```

```
                        AND   A                 lui x în A.
                        RET   Z                 Dacă A este zero, se revine
                                                pînă cînd x este deja un
                                                întreg mic.
                        CP    +81               Se compară e, exponentul, cu
                                                81 hex.
                        JR    NC,3221,T-GR-ZERO Salt dacă e este mai mare
                                                decît 80 hex.
                        LD    (HL),+00          Altfel, setare exponent pe
                        LD    A,+20             zero; se introduce 32 zecimal
                        JR    3272,NIL-BYTES    20 hex, în A si salt mai
                                                departe la NIL-BYTES pentru a
                                                face toti bitii lui x zero.
3221 T-GR-ZERO          CP    +91               Se compară e cu 91 hex, 145
                                                zecimal.
3223                    JR    NZ,323F,T-SMALL   Salt dacă e nu este 91 hex.
```

Următorii 26 octeti par a descrie testul dacă x este de fapt -65536 zecimal, adică 91 80 00 00 00, si dacă este, se setează pe 00 FF 00 00 00. Aceasta este o greseală. Cum deja s-a explicat la octetul 303B de deasupra, sistemul Spectrum nu poate trata acest număr. Aici rezulttul este simplu de făcut INT (-65536) să revină la valoarea -1. Este păcat, pînă cînd numărul nu va putea fi perfect în regulă dacă este lăsat singur. Remediul pare a fi simplu, de a omite 28 octeti din 3223 deasupra lui 323E inclusiv din program.

```
3225                    INC   HL                HL indică cei patru octeti ai
                        INC   HL                lui x, unde cei 17 biti ai
                        INC   HL                părtii întregi a lui x se
                                                termină după primul bit.
                        LD    A,+80             Primul bit este obtinut în A,
                        AND   (HL)              folosind 80 hex ca mască.
                        DEC   HL                Acest bit si cei 8 biti
                        OR    (HL)              anteriori sînt testati
                                                împreună fată de zero.
                        DEC   HL                HL indică al doilea octet din
                                                HL.
                        JR    NZ,3223,T-FIRST   Dacă încă este diferit de
                                                zero, testul se poate
                                                termina.
                        LD    A,+80             Altfel, testul pentru -65536
                        XOR   (HL)              este acum completat; 91 80
                                                00 00 00 va lăsa acum
                                                fanionul de zero setat.
3233 T-FIRST            DEC   HL                HL indică primul octet a lui
                                                x.
                        JR    NZ,326C,T-EXPNENT Dacă zero este resetat, se
                                                face saltul.
                        LD    (HL),A            Primul octet este setat pe
                                                zero.
                        INC   HL                HL indică al doilea octet.
                        LD    (HL),+FF          Al doilea octet este setat pe
                                                FF.
                        DEC   HL                HL indică din nou primul
                                                octet.
                        LD    A,+18             Ultimii 24 de biti trebuie să
                                                fie zero.
                        JR    3272,NIL-BYTES    Saltul la NIL-BYTES
                                                completează numărul 00 FF 00
                                                00 00.
```

Dacă octetul exponent al lui x este între 81 si 90 hex (129 si 144 zecimal) inclusiv, I(x) este un 'întreg mic', si va fi comprimat în unul sau doi octeti. Dar mai întîi se face un test pentru a vedea dacă x este, după toate, un număr mare.

```
323F T-SMALL            JR    NC,326D,X-LARGE   Salt cu octetul exponent 92
                                                sau mai mare (este mai bine
                                                de sărit cu 91).
                        PUSH  DE                Se salvează STKEND în DE.
                        CPL                     Intervalul 129<=A<=144 devine
                                                126<=A<=111.
                        ADD   A,+91             Intervalul este acum 15 dec>=
                                                A>=0.
                        INC   HL                HL indică al doilea octet.
                        LD    D,(HL)            Al doilea octet în D.
                        INC   HL                HL indică al treilea octet.
                        LD    E,(HL)            Al treilea octet în E.
                        DEC   HL                HL indică din nou primul
                        DEC   HL                octet.
```

```
                LD    C,+00       Se presupune un număr pozitiv
                BIT   7,D         Acum se testează dacă este
                                  negativ (bitul 7 setat).
                JR    Z,3252,T-NUMERIC  Salt dacă este pozitiv.
                DEC   C           Se schimbă semnul.
3252 T-NUMERIC  SET   7,D         Se inserează bitul adevărat
                                  numeric 1 în D.
                LD    B,+08       Acum se testează dacă A>=8
                SUB   B           (numai un octet) sau doi
                                  octeti sînt necesari.
                ADD   A,B         Se lasă A neschimbat.
                JR    C,325E,T-TEST  Salt dacă sînt necesari doi
                                  octeti.
                LD    E,D         Se pune un octet în E.
                LD    D,+00       Si se setează D pe zero.
                SUB   B           Acum 1<=A<=7 pentru a număra
                                  deplasările necesare.
325E T-TEST     JR    Z,3267,T-STORE  Salt dacă nu este necesară
                                  nici o deplasare.
                LD    B,A         B va contoriza deplasările.
3261 T-SHIFT    SRL   D           Se deplasează D si E la
                RR    E           dreapta de B ori pentru a da
                                  numărul corect.
                DJNZ  3261,T-SHIFT  Se ciclează pînă B este zero.
3267 T-STORE    CALL  2D8E,INT-STORE  Se stochează rezultatul în
                                  stivă.
                POP   DE          Se reface STKEND în DE.
                RET               Sfîrsit.
```

A rămas să se ia în considerare valorile mari pentru x.

```
326C T-EXPNENT  LD    A,(HL)      Se aduce octetul exponent al
                                  lui x în A.
326D X-LARGE    SUB   +A0         Se scade din e 160 zecimal,
                                  A0 hex.
                RET   P           Se revine pentru plus -
                                  partea întreagă a lui x este
                                  nesemnificativă.  (Dacă
                                  adevăratul exponent a fost
                                  redus la zero, 'punctul
                                  binar' va veni la sau după
                                  sfîrsitul celor patru octeti
                                  ai mantisei).
                NEG               Altfel, este negat restul;
                                  aceasta dă numărul de biti ce
                                  vor deveni zero (numărul de
                                  biti de după 'punctul
                                  binar').
```

Acum pot fi stersi bitii mantisei.

```
3272 NIL-BYTES  PUSH  DE          Se salvează valoarea curentă
                                  a lui DE (STKEND).
                EX    DE,HL       HL va indica cu unu după al
                                  cincilea octet.
                DEC   HL          Acum HL indică al cincilea
                                  octet al lui x.
                LD    B,A         Se aduce numărul bitilor ce
                SRL   B           vor fi setati pe zero în B si
                SRL   B           se împarte cu 8 pentru a da
                SRL   B           numărul tuturor bitilor
                                  implicati.
                JR    Z,3283,BITS-ZERO  Salt dacă rezultatul este
                                  zero.
327E BYTE-ZERO  LD    (HL),+00    Altfel, se setează octetii pe
                                  zero;
                DEC   HL          B îi numără.
                DJNZ  327E,BYTE-ZERO
3283 BITS-ZERO  AND   +07         Se aduce A (mod 8); aceasta
                                  este numărul de biti care mai
                                  trebuie setati pe zero.
                JR    Z,3290,IX-END  Salt la sfîrsit dacă nu mai
                                  este nimic de făcut.
                LD    B,A         B va număra biti acum.
                LD    A,+FF       Pregătire mască.
328A LESS-MASK  SLA   A           Cu fiecare buclă un zero
                DJNZ  328A,LESS-MASK  intră în mască si astfel se
                                  realizează o mască de lungime
                                  corectă.
```

```
                    AND   (HL)              Bitii nedoriti din (HL) sînt
                    LD    (HL),A            pierduti cum s-a făcut
                                            mascarea.
3290 IX-END         EX    DE,HL             Revenire indicator în HL.
                    POP   DE                Se readuce STKEND în DE.
                    RET                     Sfîrsit.
```

THE 'RE-STACK TWO' SUBROUTINE (SUBRUTINA 'RE-STACK TWO')
Această subrutină este apelată pentru a restoca doi 'întregi mici' în totalitatea celor cinci octeti ai formei în virgulă mobilă pentru operatiile binare de adunare, înmultire si împărtire. Acestea sînt realizate apelînd subrutina următoare de două ori.

```
3293 RE-ST-TWO      CALL  3296,RESTK-SUB    Apelare subrutină, si apoi se
                                            continuă în ea pentru a doua
                                            apelare.
3296 RESTK-SUB      EX    DE,HL             Se interschimbă indicatorii
                                            la fiecare apelare.
```

THE 'RE-STACK' SUBROUTINE (SUBRUTINA 'RE-STACK')
(Deplasament 3D - vezi CALCULATE după: 're-stack')

Această subrutină este apelată pentru a restoca un număr (care poate fi un 'întreg mic') în totalitatea celor cinci octeti ai formei în virgulă mobilă. Este folosită pentru un singur număr de ARCTAN si de asemenea, de-a lungul deplasamentului calculatorului, de EXP, LN si 'get-argt'.

```
3297 RE-STACK       LD    A,(HL)            Dacă primul octet nu este
                    AND   A                 zero, se revine - numărul nu
                    RET   NZ                poate fi un 'întreg mic'.
                    PUSH  DE                Se salvează 'celălalt'
                                            indicator în DE.
                    CALL  2D7F,INT-FETCH    Se aduce semnul în C si
                                            numărul în DE.
                    XOR   A                 Se sterge registrul A.
                    INC   HL                Se indică a cincea locatie.
                    LD    (HL),A            Se setează al cincelea octet
                                            pe zero.
                    DEC   HL                Se indică a patra locatie.
                    LD    (HL),A            Se setează a patra locatie pe
                                            zero: octetii 2 si 3 vor
                                            contine mantisa.
                    LD    B,+91             Se setează B pe 145 zecimal
                                            pentru exponent, adică pentru
                                            mai mult de 16 biti în
                                            întreg.
                    LD    A,D               Se testează dacă D este zero,
                    AND   A                 astfel încît să fie necesari
                                            cel mult 8 biti.
                    JR    NZ,32B1,RS-NRMLSE Salt dacă este nevoie de mai
                                            mult de 8 biti.
                    OR    E                 Acum se testează si E.
                    LD    B,D               Se salvează zero în B (va da
                                            exponent zero dacă si E este
                                            tot zero).
                    JR    Z,32BD,RS-STORE   Salt dacă E este într-adevăr
                                            zero.
                    LD    D,E               Se mută E în D (D a fost zero
                                            E nu).
                    LD    E,B               Se setează E pe zero acum.
                    LD    B,+89             Se setează B pe 137 zecimal
                                            pentru exponent - nu mai mult
                                            de 8 biti acum.
32B1 RS-NRMLSE      EX    DE,HL             Indicatorul în DE, numărul în
                                            HL.
32B2 RSTK-LOOP      DEC   B                 Se decrementează exponntul la
                                            fiecare deplasare.
                    ADD   HL,HL             Se deplasează numărul la
                                            dreapta cu o pozitie.
                    JR    NC,32B2,RSTK-LOOP Pînă cînd transportul este
                                            setat.
                    RRC   C                 Bitul de semn în fanionul de
                                            transport acum.
                    RR    H                 Este inserat în locul unde
                    RR    L                 numărul s-a deplasat înapoi
                                            cu un spatiu - acum normal.
                    EX    DE,HL             Indicatorul octetului 4
                                            înapoi în HL.
32BD RS-STORE       DEC   HL                Se indică a treia locatie.
```

```
LD      (HL),E      Se stochează al treilea octet
DEC     HL          Se indică a doua locatie.
LD      (HL),D      Se stochează al doilea octet.
DEC     HL          Se indică prima locatie.
LD      (HL),B      Se stochează octetul exponent
POP     DE          Se     readuce     'celălalt'
                    indicator în DE.
RET                 Sfîrsit.
```

THE FLOATING-POINT CALCULATOR (DISPOZITIVUL DE CALCUL IN VIRGULA MOBILA)

THE TABLE OF CONSTANTS (TABELUL DE CONSTANTE)

Acest prim tabel contine cele mai folosite si frecvent necesare cinci numere zero, unu, jumătate, jumătatea lui pi si zece. Numerele sînt continute într-o formă condensată are este extinsă cu subrutina STACK LITERALS, vezi mai jos, pentru a da forma în virgulă mobilă cerută.

		data		constanta	prin extensie se obtine: exp.mantixa: ((Hex.)
32C5	stk-zero	DEFB	+00	zero	00 00 00 00 00
		DEFB	+B0		
		DEFB	+00		
32C8	stk-one	DEFB	+40	unu	00 00 01 00 00
		DEFB	+B0		
		DEFB	+00		
		DEFB	+01		
32CC	stk-half	DEFB	+30	jumatate	80 00 00 00 00
		DEFB	+00		
32CE	stk-pi/2	DEFB	+F1	jumătatea lui pi	81 49 0F DA A2
		DEFB	+49		
		DEFB	+0F		
		DEFB	+DA		
		DEFB	+A2		
32D3	stk-ten	DEFB	+40	zece	00 00 0A 00 00
		DEFB	+B0		
		DEFB	+00		
		DEFB	+0A		

THE TABLE OF ADDRESSES (TABELUL DE ADRESE)

Acest al doilea tabel este un tabel de căutare a adreselor celor 66 de subrutine operationale ale calculatorului. Deplasamentele folosite la indexarea din tabel provin fie din codurile operatiilor folosite în SCANNING, vezi 2734, etc., fie din literalurile care urmează instructia RST 0028.

	deplasament	etichetă	adresă		deplasament	etichetă	adresă
32D7	00	jump-true	8F 36	3319	21	tan	DA 37
32D9	01	exchange	3C 34	331B	22	asn	33 38
32DB	02	delete	A1 33	331D	23	acs	43 38
32DD	03	subtract	0F 30	331F	24	atn	E2 37
21DF	04	multiply	CA 30	3321	25	ln	13 37
32E1	05	division	AF 31	3323	26	exp	C4 36
32E3	06	to-power	51 38	3325	27	int	AF 36
32E5	07	or	1B 35	3327	28	sqr	4A 38
32E7	08	no-&-no	24 35	3329	29	sgn	92 34
32E9	09	no-l-eql	3B 35	332B	2A	abs	4A 34
32EB	0A	no-gr-eq	3B 35	332D	2B	peek	A0 34
32ED	0B	nos-neql	3B 35	332F	2C	in	A5 34
32EF	0C	no-grtr	3B 35	3331	2D	usr-no	B3 34
32F1	0D	no-less	3B 35	3333	2E	str$	1F 36
32F3	0E	nos-eql	3B 35	3335	2F	chr$	B9 35
32F5	0F	addition	14 30	3337	30	not	01 35
32F7	10	str-&-no	2D 35	3339	31	duplicate	C0 33

32F9	11	str-l-eql	38 35	3339	32	n-mod-m	A0 3A
32FB	12	str-gr-eq	38 35	333B	33	jump	86 36
32FD	13	strs-neql	38 35	333F	34	stk-data	C4 33
32FF	14	str-grtr	38 35	3341	35	dec-jr-nz	7A 34
3301	15	str-less	38 35	3343	36	less-0	06 35
3303	16	strs-eql	38 35	3345	37	greater-0	F9 34
3305	17	strs-add	9C 35	3347	38	end-calc	9B 36
3307	18	val$	DE 35	3349	39	get-argt	83 37
3309	19	usr-$	BC 34	334B	3A	truncate	14 32
330B	1A	read-in	45 36	334D	3B	fp-calc-2	A2 33
330D	1B	negate	6E 34	334F	3C	e-to-fp	4F 2D
330F	1C	code	69 36	3351	3D	re-stack	97 32
3311	1D	val	DE 35	3353	3E	series-06 etc.	49 34
3313	1E	len	74 36	3355	3F	stk-zero etc.	1B 34
3315	1F	sin	B5 37	3357	40	st-mem-0 etc.	2D 34
3317	20	cos	AA 37	3359	41	get-mem-0 etc.	0F 34

Notă: Ultimele patru subrutine sînt subrutine multifunctionale si sînt introduse cu un parametru care este o copie a celor cinci biti din partea dreaptă a literalului originar. Urmează prezentarea întregului set:

Deplasament 3E: series-06,series-08,& series-0C; literaluri 86, 88 & 8C.

Deplasament 36: stk-zero,stk-one,stk-half,stk-pi/2 & stk-ten; literaluri A0 la A4.

Deplasament 40: st-mem-0,st-mem-1,st-mem-2,st-mem-3,st-mem-4 & st-mem-5; literaluri C0 la C5.

Deplasament 41: get-mem-0,get-mem-1,get-mem-2,get-mem-3,get-mem-4 & get-mem-5; literaluri E0 la E5

THE 'CALCULATE' SUBROUTINE (SUBRUTINA 'CALCULATE')

Această subrutină este folosită pentru a realiza calculele în virgulă mobilă. Acestea pot fi considerate ca fiind de trei tipuri:

i. Operatii binare, cum ar fi adunarea, unde două numere în forma cu virgulă mobilă sînt adunate împreună pentru a da o 'valoare ultimă'.

ii. Operatii unare, cum ar fi sin, unde 'ultima valoare' este schimbată pentru a da rezultatul corespunzător functiei ca o nouă 'ultimă valoare'.

iii. Operatii de manevră, cum ar fi st-mem-0, unde 'ultima valoare' este copiată în primii cinci octeti din zona de memorie a calculatorului.

Operatiile ce trebuie efectuate sînt specificate ca o serie de octeti de informatii, literalurile, care urmează o instructie RST 0028 care apelează această subrutină. Ultimul literal din listă este întotdeauna '38', care conduce la terminarea întregii operatii.

In cazul în care trebuie efectuată o singură operatie, deplasamentul operatiei se poate trece în CALCULATOR în registrul B, si se efectuează operatia '3B', operatia SINGLE CALCULATION.

De asemenea, mai este posibil să se apeleze această subrutină recursiv, adică se ciclează pe însăsi, si într-un asemenea caz este posibilă utilizarea variabilei sistem BREG ca un numărător care controlează cîte operatii sînt efectuate înainte de revenire.

Prima parte a acestei subrutine este complicată, dar în esentă ea realizează cele două taskuri pentru setarea registrilor astfel încît ei să conțină valorile cerute, si pentru a produce un deplasament si probabil un parametru, dintr-un literal, care a fost considerat în mod curent.

Deplasamentul este folosit pentru indexarea tabelului de adrese al calculatorului, vezi deasupra, pentru a găsi subrutina de adresă cerută.

Parametrul se foloseste cînd sînt apelate subrutinele multifunctionale.

Notă: Un număr în virgulă mobilă poate fi în realitate un set de parametrii

sir.

```
335B CALCULATE    CALL  35BF,STK-PNTRS    Se presupune o operatie unară
                                          si de aceea se setează HL
                                          pentru a indica începutul
                                          'ultimei valori' din stiva
                                          calculatorului si DE unul
                                          după acest număr în virgulă
                                          mobilă (STKEND).
335E GEN-ENT-1    LD    A,B               Sau se transferă temporar
                  LD    (BREG),A          deplasmentul unei operatii
                                          singulare în BREG sau, cînd
                                          se foloseste subrutina
                                          recursiv, se trece în BREG
                                          parametrul ce va fifolosit ca
                                          si contor.
3362 GEN-ENT-2    EXX                     Adresa de revenire a
                  EX    (SP),HL           subrutinei se stochează în
                  EXX                     H'L'. Acesta salvează
                                          indicatorul primului literal.
                                          Intrarea în CALCULATOR prin
                                          GEN-ENT-2 este folosită cînd
                                          BREG este folosit ca si
                                          contor si nu trebuie
                                          conturbat.
3365 RE-ENTRY     LD    (STKEND),DE       Acum se introduce o buclă
                                          pentru a trata fiecare
                                          literal din lista care
                                          urmează instructiunea
                                          apelantă; asa că mai întîi,
                                          întotdeauna, se setează
                                          STKEND.
                  EXX                     Se trece la setul de
                  LD    A,(HL)            registrii auxiliari si se
                                          aduce literalul pentru
                                          această buclă.
                  INC   HL                H'L' va indica următorul
                                          literal.
336C SCAN-ENT     PUSH  HL                Acest indicator este salvat
                                          pentru scurt timp în stiva
                                          masinii. SCAN-ENT este
                                          folosit de subrutina SINGLE
                                          CALCULATION pentru a afla
                                          subrutina care este cerută.
                  AND   A                 Se testează registrul A.
                  JP    P,3380,FIRST-3D   Se separă literalul simplu de
                                          literalurile multifunctionale
                                          Salt pentru literalurile 00 -
                                          3D.
                  LD    D,A               Salvare literal în D.
                  AND   +60               Se continuă doar cu bitii 5 &
                                          6.
                  RRCA                    Patru deplasări la dreapta îi
                  RRCA                    fac acum să fie bitii 1 & 2.
                  RRCA
                  RRCA
                  ADD   A,+7C             Deplasamentele cerute sînt 3E
                  LD    L,A               - 41 si L va contine acum
                                          copia deplasamentului cerut.
                  LD    A,D               Acum se realizează parametrul
                  AND   +1F               luînd bitii 1, 2, 3 & 4 ai
                                          literalului; se păstrează
                                          parametrul în A.
                  JR    338E,ENT-TABLE    Salt înainte pentru a
                                          determina adresa subrutinei
                                          cerute.
3380 FIRST-3D     CP    +18               Salt înainte dacă se
                  JR    NC,338C,DOUBLE-A  efctuează o operatie unară.
                  EXX                     Toate subrutinele care
                  LD    BC,+FFFB          efectuează operatii binare
                  LD    D,H               solicită ca HL să indice
                  LD    E,L               primul operand si DE să
                  ADD   HL,BC             indice al doilea operand
                  EXX                     ('ultima valoare'), asa cum
                                          apar in stiva calculatorului.
338C DOUBLE-A     RLCA                    Asa cum fiecare intrare în
                  LD    L,A               tabelul de adrese necesită
                                          doi octeti, deplasamentul
                                          realizat este dublat.
```

```
338E ENT-TABLE   LD    DE,+32D7      Adresa de bază a tabelului.
                 LD    H,+00         Adresa intrării tabelului
                 ADD   HL,DE         cerut este formată în HL; iar
                 LD    E,(HL)        adresa subrutinei cerute este
                 INC   HL            încărcată în registrul
                 LD    D,(HL)        pereche DE.
                 LD    HL,+3365      Adresa RE-ENTRY la 3365 este
                 EX    (SP),HL       pusă în stiva masinii sub
                 PUSH  DE            adresa subrutinei.
                 EXX                 Revenire la setul de
                                     registrii principali.
                 LD    BC,(STKEND-hi) Valoarea curentă a lui BREG
                                     este transferată în registrul
                                     B astfel returnînd
                                     deplasamentul operatiei
                                     singulare. (Vezi COMPARISON
                                     la 353B.)
33A1 delete      RET                 Salt indirect la subrutina
                                     cerută.
```

THE 'DELETE' SUBROUTINE (SUBRUTINA 'DELETE' (stergere))
(Deplasament 02: 'delete')

Această subrutină contine doar instructia singulară RET la 33A1, deasupra. Literalul '02' rezultat în această subrutină a fost considerat ca o operatie binară care trebuie introdusă cu un prim număr adresat de registrul pereche HL si un al doilea număr adresat de registrul pereche DE si rezultatul obtinut este din nou adresat de registrul pereche HL.

Instructia singulară RET astfel conduce la primul număr ce a fost considerat ca 'ultima valoare' rezultantă si al doilea număr se consideră ca fiind sters. Bineînteles că numărul nu a fost sters din memorie, dar rămîne inactiv si probabil că în curînd va fi supraimprimat.

THE 'SINGLE OPERATION' SUBROUTINE (SUBRUTINA 'OPERATIE SINGULARA')
(Deplasament 3B: 'fp-calc-2')

Această subrutină este apelată numai din SCANNING la 2757 hex si este folosită pentru a efectua o operatie aritmetică singulară. Deplasamentul care specifică ce operatie trebuie efectuată este înlocuit în calculator în registrul B si transferat ulterior în variabila sistem BREG.

Efectul apelării acestei subrutine este în esentă de a executa un salt la subrutina corespunzătoare pentru operatia singulară.

```
33A2 fp-calc-2   POP   AF            Se înlătură adresa RE-ENTRY.
                 LD    A,(BREG)      Se transferă deplasamentul în
                                     A.
                 EXX                 Se introduce setul de
                                     registrii auxiliari.
                 JR    336C,SCAN-ENT Salt înapoi pentru a găsi
                                     adresa cerută; se stochează
                                     adresa RE-ENTRY si salt la
                                     subrutina pentru operatie.
```

THE 'TEST 5-SPACES' SUBROUTINE (SUBRUTINA 'TESTARE 5 SPATII')
Această subrutină testează dacă în memorie există suficient spatiu pentru alti cinci octeti de număr în virgulă mobilă care să fie adăugati în stiva calculatorului.

```
33A9 TEST-5-SP   PUSH  DE            Se adresează DE pentru scurt
                                     timp.
                 PUSH  HL            Se salvează HL pentru scurt
                                     timp.
                 LD    BC,+0005      Se specifică faptul că testul
                                     se face pentru 5 octeti.
                 CALL  1F05,TEST-ROOM Se execută testul.
                 POP   HL            Se readuce HL.
                 POP   DE            Se readuce DE.
                 RET                 Sfîrsit.
```

THE 'STACK NUMBER' SUBROUTINE (SUBRUTINA 'STOCARE NUMAR')
Această subrutină este apelată de BEEP si de SCANNING de două ori pentru a copia STKEND în DE, pentru a muta numărul în virgulă mobilă în stiva calculatorului si pentru a reseta STKEND din DE. Pentru a face mutarea actuală se apelează 'MOVE-FP'.

```
33B4 STACK-NUM   LD    DE,(STKEND)   Se copiază STKEND în DE ca
                                     adresa destinatiei.
                 CALL  33C0,MOVE-FP  Se mută numărul.
                 LD    (STKEND),DE   Se resetează STKEND din DE.
```

```
                RET                 Sfîrsit.
```

THE 'MOVE A FLOATING-POINT NUMBER' SUBROUTINE (SUBRUTINA 'MUTARE NUMAR IN VIRGULA MOBILA')
(Deplasament 31; 'duplicate')

Această subrutină mută un număr în virgulă mobilă în vîrful stivei calculatorului (3 cazuri) sau din vîrful stivei în spatiul memoriei calculatorului (1 caz). Este de asemenea apelată în calculator cînd pur si simplu se dublează numărul din vîrful stivei calculatorului, 'ultima valoare' astfel extinzînd stiva cu cinci octeti.

```
33C0 MOVE-FP    CALL  33A9,TEST-5-SP      Se face un test pentru spatiu
                LDIR                      Se mută cei cinci octeti
                                          implicati.
                RET                       Sfîrsit.
```

THE 'STACK LITERALS' SUBROUTINE (SUBRUTINA 'STOCARE LITERALURI')
(Deplasament 34; 'stk-data')

Această subrutină plasează în stiva calculatorului ca o 'ultimă valoare' numărul în virgulă mobilă ce o înlocuieste ca 2, 3, 4 sau 5 literaluri.

Cînd este apelată folosind deplasamentul '34' literalul urmează '34' în lista de literaluri; cînd este apelată de SERIES GENERATOR, vezi în continuare, literalurile sînt înlocuite de subrutina care apelează pentru o serie care trebuie generată; iar cînd este apelată de SKIP CONSTANTS & STACK A CONSTANT literalurile se obtin din tabelul de constante al calculatorului (32C5 - 32D6).

In fiecare caz, primul literal înlocuit este divizat de 40 hex si cîtul întreg plus 1 determină dacă 1, 2, 3 sau 4 literaluri în continuare vor fi luate din sursă pentru a forma mantisa numărului. Oricare octeti incompleti din cei cinci octeti care vor forma un număr în virgulă mobilă pe cinci octeti sînt setati pe zero. Primul literal este de asemenea folosit pentru a se determina exponentul, după reducerea mod. 40 hex., în afară de cazul cînd restul este zero, caz în care se foloseste al doilea literal, asa cum se găseste el, fără reducere în mod 40 hex. In fiecare caz, se adaugă 50 hex la literal, rezultînd octetul exponent mărit, e (adevăratul exponent e' plus 80 hex.). Restul din cei cinci octeti sînt stocati, incluzînd orice zerouri necesare, si subrutina revine.

```
33C6 STK-DATA   LD    H,D
                LD    L,E                 Această subrutină realizează
                                          operatia de manevră a
                                          adăugării unei ultime valori
                                          în stiva calculatorului; deci
                                          HL este setat pentru a indica
                                          cu un număr după ultima
                                          valoare prezentă si deci
                                          indică rezultatul.
33C8 STK-CONST  CALL  33A9,TEST-5-SP       Se testează dacă există
                                          într-adevar loc.
                EXX                       Se trece la setul registrilor
                PUSH  HL                  auxiliari si se stochează
                EXX                       indicatorul pentru următorul
                                          literal.
                EX    (SP),HL             Se schimbă indicatorul
                                          rezultatului si indicatorul
                                          următorului literal.
                PUSH  BC                  Se salvează BC pentru scurt
                                          timp.
                LD    A,(HL)              Primul literal este pus în A
                AND   +C0                 si împărtit cu 40 hex pentru
                RLCA                      a da valorile întregi 0, 1,
                RLCA                      2 sau 3.
                LD    C,A                 Valoarea întreagă este
                INC   C                   transferată în C si
                                          incrementată, dînd astfel
                                          ordinul 1, 2, 3 sau 4
                                          numărului de literaluri care
                                          va fi necesar.
                LD    A,(HL)              Literalul este adus din nou;
                AND   +3F                 reducere mod 40 hex. si
                JR    NZ,33DE,FORM-EXP    înlăturare ca necorespunzător
                INC   HL                  dacă restul este zero; caz în
                LD    A,(HL)              care următorul literal este
                                          adus si utilizat neredus.
33DE FORM-EXP   ADD   A,+50               Exponentul, e, este format
                LD    (DE),A              prin adunarea cu 50 hex. si
                                          este trecut în stiva
                                          calculatorului ca fiind
```

```
                                          primul din cei cinci octeti
                                          ai rezultatului.
                LD    A,+05               Numărul de literaluri
                SUB   C                   specificat în C este luat din
                INC   HL                  sursă si introdus în octetii
                INC   DE                  rezultatului.
                LD    B,+00
                LDIR
                POP   BC                  Se restituie BC.
                EX    (SP),HL             Se readuce indicatorul
                EXX                       rezultatului în HL si
                                          indicatorul următorului
                                          literal în pozitia sa
                                          obisnuită în H' & L'.
                POP   HL
                EXX
                LD    B,A                 Numărul cerut de octeti zero
                XOR   A                   în această fază este dat de
33F1 STK-ZEROS  DEC   B                   5-C-1; si acest număr de
                RET   Z                   zerouri este adăugat
                LD    (DE),A              rezultatului pentru a forma
                                          cei cinci octeti ceruti.
                INC   DE
                JR    33F1,STK-ZEROS
```

THE 'SKIP CONSTANTS' SUBROUTINE (SUBRUTINA 'OMITERE CONSTANTE')
Aceasta subrutină este introdusă cu registrul pereche HL continînd adresa de bază a tabelului de constante a calculatorului si cu registrul A continînd un parametru care arată care dintre cele cinci constante a fost cerută.
Subrutina efectuează operatia invalidă de încărcare a celor cinci octeti pentru fiecare constantă nedorită în locatiile 0000, 0001, 0002, 0003 si 0004 la începutul lui ROM înă cînd constanta cerută este găsită.
Subrutina revine cu registrul pereche HL continînd adresa de bază a constantei cerute în cadrul tabelului de constante.

```
33F7 SKIP-CONS  AND   A                   Revenire subrutină dacă
33F8 SKIP-NEXT  RET   Z                   parametrul a fost zero sau
                                          cînd constanta cerută a fost
                                          găsită.
                PUSH  AF                  Salvare parametru.
                PUSH  DE                  Salvare indicator rezultat.
                LD    DE,+0000            Inlocuire adresă.
                CALL  33C8,STK-CONST      Se efectuează o stocare
                                          imaginară a unei constante
                                          dezvoltate.
                POP   DE                  Se readuce indicatorul
                                          rezultatului.
                POP   AF                  Se readuce parametrul.
                DEC   A                   Contorizare bucle.
                JR    33F8,SKIP-NEXT      Salt înapoi pentru a
                                          considera valoarea
                                          contorului.
```

THE 'MEMORY LOCATION' SUBROUTINE (SUBRUTINA 'LOCATIE DE MEMORIE')
Această subrutină găseste adresa de bază pentru fiecare portiune de cinci octeti din spatiul de memorie al calculatorului în care sau din care un număr în virgulă mobilă trebuie mutat din sau în stiva calculatorului. Ea realizează această operatie adăugînd de cinci ori parametrul înlocuit la adresa de bază a zonei care este continută în registrul pereche HL.
De notat că atunci cînd o variabilă FOR-NEXT a fost tratată, atunci indicatorii sînt schimbati astfel încît variabila este tratată ca si cînd ea ar fi fost în zona de memorie a calculatorului (vezi adresa 1D20).

```
3406 LOC-MEM    LD    C,A                 Se copiază parametrul în C.
                RLCA                      Se dublează parametrul.
                RLCA                      Se dublează acest rezultat.
                ADD   A,C                 S adună valoarea parametrului
                                          pentru a obtine de cinci ori
                                          valoarea initială.
                LD    C,A                 Acest rezultat este dorit în
                LD    B,+00               registrul pereche BC.
                ADD   HL,BC               Se produce noua adresă de
                                          bază.
                RET                       Sfîrsit.
```

THE 'GET FROM MEMORY AREA' SUBROUTINE (SUBRUTINA 'ADUCERE DIN ZONA DE MEMORIE')
(Deplasamente C0 la C5: 'st-mem-0' la 'st-mem-5')

Această subrutină este apelată folosind literalurile E0 la E5, iar parametrul provenit din aceste literaluri este continut în registrul A. Subrutina apelează MEMORY LOCATION pentru a pune adresa sursă cerută în registrul pereche HL si MOVE A FLOATING-POINT NUMBER pentru a copia cei cinci octeti implicati din spatiul de memorie al calculatorului în vîrful stivei calculatorului pentru a forma o nouă 'valoare ultimă'.

```
340F get-mem-0      PUSH  DE                Salvare indicator rezultat.
     etc.           LD    HL,(MEM)          Se aduc indicatorul în zona
                                            curentă de memorie (vezi
                                            deasupra).
                    CALL  3406,LOC-MEM      S-a găsit adresa de bază.
                    CALL  33C0,MOVE-FP      Cei cinci octeti sînt mutati.
                    POP   HL                Setare indicator rezultat.
                    RET                     Sfîrsit.
```

THE 'STACK A CONSTANT' SUBROUTINE (SUBRUTINA 'STOCARE CONSTANTA')
(Deplasamente A0 la A4: 'stk-zero', 'stk-one', stk-half', 'stk-pi/2' & 'stk-ten')

Această subrutină foloseste SKIP CONSTANTS pentru a găsi adresa de bază a constantelor cerute din tabelul de constante a calculatorului si apoi apelează STACK LITERALS, introdus pe la STK-CONST, pentru a realiza forma extinsă a constantei 'ultima valoare' din stiva calculatorului.

```
341B stk-zero       LD    H,D               Se setează HL pentru a
     etc.                                   contine indicatorul rezultat.
                    LD    L,E
                    EXX                     Se trece la setul de
                    PUSH  HL                registrii auxiliari si se
                                            salvează următorul indicator
                                            al literalului.
                    LD    HL,+32C5          Adresa de bază a tabelului de
                                            constante a calculatorului.
                    EXX                     Înapoi la setul principal de
                                            registrii.
                    CALL  33F7,SKIP-CONS    Se găseste adresa de bază
                                            cerută.
                    CALL  33C8,STK-CONST    Extindere constantă.
                    EXX
                    POP   HL                Se readuce următorul
                                            indicator literal.
                    EXX
                    RET                     Sfîrsit.
```

THE 'STORE IN MEMORY AREA' SUBROUTINE (SUBRUTINA 'STOCARE ÎN ZONA DE MEMORIE')
(Deplasamente de la C0 la C5: 'st-mem-0' la 'st-mem-5')

Această subrutină este apelată folosind literalurile de la C0 la C5 iar parametrul provenit din aceste literaluri este continut în registrul A. Această subrutină este foarte asemănătoare cu subrutina GET FROM MEMORY, dar indicatorii sursă si destinatie sînt schimbati.

```
342D st-mem-0       PUSH  HL                Salvare indicator rezultat.
     etc.           EX    DE,HL             Sursa în DE pentru scurt timp
                    LD    HL,(MEM)          Se aduce indicatorul în zona
                                            curentă de memorie.
                    CALL  3406,LOC-MEM      S-a găsit adresa de bază.
                    EX    DE,HL             Se interschimbă indicatorii
                                            sursă si destinatie.
                    CALL  33C0,MOVE-FP      Cei cinci octeti sînt
                                            transferati.
                    EX    DE,HL             'Ultima valoare' +5, adică
                                            STKEND, în DE.
                    POP   HL                Indicatorul rezultat în HL.
                    RET                     Sfîrsit.
```

De notat că indicatorii HL si DE rămîn asa cum au fost, indicînd STKEND-5 si respectiv STKEND, astfel încît 'ultima valoare' rămîne în stiva calculatorului. Dacă se cere, aceasta poate fi îndepărtată prin utilizarea lui 'delete'.

THE ' EXCHANGE' SUBROUTINE (SUBRUTINA 'INTERSCHIMBARE')
(Deplasament 01: 'exchange')

Această operatie binară 'interschimbă' primul număr cu al doilea număr, aceasta însemnînd că cele două numere din vîrful stivei calculatorului sînt interschimbate.

```
343C EXCHANGE   LD    B,+05          Sînt implicati cinci octeti,
343E SWAP-BYTE  LD    A,(DE)         Fiecare octet al celui de al
                                     doilea număr.
                LD    C,(HL)         Fiecare octet al primului
                                     număr.
                EX    DE,HL          Se schimbă sursa si
                                     destinatia.
                LD    (DE),A         Acum la primul număr.
                LD    (HL),C         Acum la al doilea număr.
                INC   HL             Se trece la considerarea
                INC   DE             următoarei perechi de octeti.
                DJNZ  343E,SWAP-BYTE Se interschimbă cei cinci
                                     octeti.
                EX    DE,HL          Se corectează indicatorii ca
                                     si cum numărul 5 este un
                                     număr impar.
                RET                  Sfîrsit.
```

THE 'SERIES GENERATOR' SUBROUTINE (SUBRUTINA 'GENERATOR SERII')
(Deplasamentele 86, 88 & 8C: 'series-06', 'series-08' & 'series-0C')

Această subrutină importantă generează seriile polinoamelor lui Cebîsev, care se folosesc la aproximarea lui SIN, ATN, LN si EXP si de aici la derivarea altor functii aritmetice care depind de acestea (COS, TAN, ASN, ACS, ** si SQR).

Polinoamele sînt generate, de la n = 1,2,..., prin relatia de recurentă:

$T_{n+1}(z) = 2zT_n(z) - T_{n-1}(z)$ unde $T_n(z)$ este al n-lea polinom Cebîsev în z.

Seriile de fapt generează:

$T_0, 2T_1, 2T_2, \ldots, 2T_{n-1}$, unde n este 6 pentru SIN, 8 pentru EXP si 12 zecimal pentru LN si ATN.

Coeficientii puterilor lui z din aceste polinoame pot fi găsiti în "Handbook of Mathematical Functions" de M.Abramowitzs I.A.Stegun (Dover 1965), pag.795.

Programele BASIC care arată generarea fiecăreia din cele patru functii este dată în Anexa.

In termeni simplii, această subrutină este apelată cu 'ultima valoare' în stiva calculatorului, numită Z, fiind un număr care duce la o legătură simplă cu argumentul, numit X, cînd taskul trebuie să evalueze, de exemplu, SIN X. Subrutina apelantă de asemenea furnizează lista constantelor care vor fi cerute (sase constante pentru SIN). SERIES GENERATOR manevrează apoi informatiile sale si revine la rutina apelantă ca o 'ultimă valoare' care duce la o simplă legătură cu functia cerută, de exemplu, SIN X.

Se poate considera că această subrutină are patru părti mari:

i. Setarea contorului buclei:

Subrutina apelantă trece parametrii săi în registrul A pentru utilizarea lui ca si contor. Intrarea se face pe la GEN-ENT-1, asa că se poate seta contorul.

```
3449 series-06  LD    B,A            Se mută parametrul în B.
     etc.       CALL  335E,GEN-ENT-1 De fapt o instructiune RST
                                     0028 setează contorul.
```

ii. Tratarea 'ultimei valori', Z:

Bucla generatorului cere plasarea lui 2*Z în mem-0, zero să fie plasat în mem-2 si 'ultima valoare' să fie zero.

```
                                     stiva calculatorului
                DEFB  +31,duplicate  Z,Z
                DEFB  +0F,addition   2*Z
                DEFB  +C0,st-mem-0   2*Z      mem-0 contine 2*Z
                DEFB  +02,delete     -
                DEFB  +A0,stk-zero   0
                DEFB  +C2,st-mem-2   0        mem-2 contine 0
```

iii. Bucla principală:

Seria este generată prin ciclare, folosind BREG ca si contor; constantele din subrutina apelantă se stochează pe rînd apelînd STK-DATA; calculatorul intră din nou prin GEN-ENT-2, asa că nu trebuie conturbată valoarea lui BREG; si seria este construită în forma:

$B(R) = 2*Z*B(R-1) - B(R-2) + A(R)$, pentru $R = 1,2,\ldots,N$,

unde $A(1), A(2), \ldots, A(N)$ sînt constantele înlocuite de rutina apelantă (SIN, ATN, LN si EXP) si $B(0) = 0 = B(-1)$.

A (R+1)-a buclă începe cu B(R) în stivă si cu 2*Z,B(R-2) si B(R-1) în mem-0, mem-1 si mem-2 respectiv.

```
3453 G-LOOP     DEFB  +31,duplicate  B(R),B(R)
```

```
                DEFB  +E0,get-mem-0          B(R),B(R)e2*Z
                DEFB  +E2,get-mem-2          B(R),2*B(R)*Z,B(R-1)
                DEFB  +C1,st-mem-1           mem-1 contine B(R-1)
                DEFB  +03,subtract           B(R),2*B(R)*Z-B(R-1)
                DEFB  +38,end-calc
```

Se pune în stiva calculatorului următoarea constantă.

```
                CALL  33C6,STK-DATA          B(R),2*B(R)*Z-B(R-1),A(R+1)
```

Calculatorul reintră fără a-l conturba pe BREG.

```
                CALL  3362,GEN-ENT-2
                DEFB  +0F,addition           B(R),2*B(R)*Z-B(R-1)+A(R+1)
                DEFB  +01,exchange           2*B(R)*Z-B(R-1)+A(R+1),B(R)
                DEFB  +C2,st-mem-2           mem-2 contine B(R)
                DEFB  +02,delete             2*B(R)*Z-B(R-1)+A(R!1)=B(R!1)
                DEFB  +35,dec-jr-nz          B(R+1)
                DEFB  +EE,to3453,G-LOOP
```

iv. Scăderea lui B(N-2):
Bucla de deasupra lasă B(N) în stivă si rezultatul cerut este dat de B(N)-B(N-2).

```
                DEFB  +E1,get-mem-1          B(N),B(N-2)
                DEFB  +03,subtract           B(N)-B(N-2)
                DEFB  +38,end-calc
                RET                          Sfîrsit.
```

THE 'ABSOLUTE MAGNITUDE' FUNCTION (FUNCTIA 'MARIME ABSOLUTA')
(Deplasament 2A: 'abs')

Această subrutină efectuează operatia unară prin asigurarea că bitul de semn al numărului în virgulă mobilă este resetat.
'Intregii mici' trebuie tratati separat. Cea mai mare parte a muncii se referă la operatia 'minus unar'.

```
346A abs        LD    B,+FF                  B este setat pe FF.
                JR    3474,NEG-TEST          Saltul se face la 'minus
                                             unar'.
```

THE 'UNARY MINUS' OPERATION (OPERATIA 'MINUS UNAR')
(Deplasament 1B: 'negate')

Această subrutină efectuează operatia sa unară prin schimbarea semnlui 'ultimei valori' din stiva calculatorului.
Zero este returnat pur si simplu neschimbat. Toti cei cinci octeti ai numărului în virgulă mobilă au bitul de semn modificat astfel încît în final va fi resetat (pentru 'abs') sau schimbat (pentru 'negate'). 'Intregii mici' au octetul lor de semn setat pe zero (pentru 'abs') sau schimbat (pentru 'negate').

```
346E NEGATE     CALL  34E9,TEST-ZERO         Dacă numărul este zero,
                RET   C                      subrutina revine lăsînd
                                             00 00 00 00 00 neschimbat.
                LD    B,+00                  B este setat pe +00 hex
                                             pentru 'negate'.
```

Aici se introduce 'ABS'.

```
3474 NEG-TEST   LD    A,(HL)                 Dacă primul octet este zero,
                AND   A                      se face saltul pentru a lucra
                JR    Z,3483,INT-CASE        cu 'întregi mici'.
                INC   HL                     Se indică al doilea octet.
                LD    A,B                    Se aduce +FF pentru 'abs',
                                             +00 pentru 'negate'.
                AND   +80                    Acum +80 pentru 'abs', +00
                                             pentru 'negate'.
                OR    (HL)                   Aceasta setează bitul 7
                                             pentru 'abs', dar nu modifică
                                             nimic pentru 'negate'.
                RLA                          Acum bitul 7 este schimbat,
                CCF                          ceea ce conduce la resetarea
                RRA                          bitului 7 din octetul 2
                                             pentru 'negate'.
                LD    (HL),A                 Noul octet secund este
                                             stocat.
                DEC   HL                     HL indică din nou primul
```

```
                                                    octet.
                RET                                 Sfîrsit.
```

'Cazul întregilor' dă o operatie similară cu octetul semn.

```
3483 INT-CASE   PUSH  DE                            Se salvează STKEND în DE.
                PUSH  HL                            Se salvează indicatorul
                                                    numărului în HL.
                CALL  2D7F,INT-FETCH                Se aduce semnul în C, numărul
                                                    în DE.
                POP   HL                            Se readuce indicatorul
                                                    numărului în HL.
                LD    A,B                           Se aduce +FF pentru 'abs',
                                                    +00 pentru 'negate'.
                OR    C                             Acum +FF pentru 'abs', nici o
                                                    schimbare pentru 'negate'.
                CPL                                 Acum +00 pentru 'abs' si o
                LD    C,A                           schimbare octet pentru
                                                    'negate': stocare în C.
                CALL  2D8E,INT-STORE                Se stochează rezultatul în
                                                    stivă.
                POP   DE                            Se readuce STKEND în DE.
                RET                                 Sfîrsit.
```

THE 'SIGNUN' FUNCTION (FUNCTIA 'SEMN')
(Deplasament 29: 'sgn')

Această subrutină tratează functia SGN X si de aceea redă o 'ultimă valoare' 1 dacă X este pozitiv, zero dacă X este zero si -1 dacă X este negativ.

```
3492 sgn        CALL  34E9,TEST-ZERO                Dacă X este zero, doar se
                RET   C                             revine cu zero că 'ultima
                                                    valoare'.
                PUSH  DE                            Se salvează indicatorul
                                                    pentru STKEND.
                LD    DE,+0001                      Se stochează 1 în DE.
                INC   HL                            Se indică al doilea octet al
                                                    lui X.
                RL    (HL)                          Se roteste bitul 7 în
                                                    fanionul de transport.
                DEC   HL                            Se indică din nou destinatia.
                SBC   A,A                           Se setează C pe zero dacă X
                LD    C,A                           este pozitiv si pe FF dacă X
                                                    este negativ.
                CALL  2D8E,INT-STORE                Se stochează 1 sau -1, după
                                                    cum este cazul.
                POP   DE                            Se readuce indicatorul pentru
                                                    STKEND.
                RET                                 Sfîrsit.
```

THE 'IN' FUNCTION (FUNCTIA 'IN')
(Deplasament 2C: 'in')

Această subrutină tratează functia IN X. Aceasta intră la nivelul procesorului prin portul X încărcînd BC cu X si executînd instructiunea IN A,(C).

```
34A5 in         CALL  1E99,FIND-INT2                'Ultima valoare', X, este
                                                    comprimată în BC.
                IN    A,(C)                         Semnalul este receptionat.
                JR    34B0,IN-PK-STK                Salt pentru stocarea
                                                    rezultatului.
```

THE 'PEEK' FUNCTION (FUNCTIA 'PEEK')
(Deplasament 2B: 'peek')

Această subrutină tratează functia PEEK X. 'Ultima valoare' este scoasă din stivă apelînd FIND-INT2 si este înlocuită cu valoarea continutului locatiei cerute.

```
34AC peek       CALL  1E99,FIND-INT2                Se evaluează 'ultima valoare'
                                                    rotunjită la cel mai apropiat
                                                    întreg; se testează dacă se
                                                    află în interval si se
                                                    returnează în BC.
                LD    A,(BC)                        Se aduce octetul cerut.
34B0 IN-PK-STK  JP    2D28,STACK-A                  Iesire prin salt la STACK-A.
```

THE 'USR' FUNCTION (FUNCTIA 'USR')
(Deplasament 2D: 'usr-no')

Această subrutină ('USR număr' ca formă distinctă de 'USR sir') tratează functia USR X, unde X este un număr. Valoarea lui X este obtinută în C, o adresă de revenire este stocată si codul mașină este executat din locatia X.

```
3483 usr-no     CALL  1E99,FIND-INT2      Se evaluează 'ultima valoare'
                                          rotunjită la cel mai apropiat
                                          întreg; se testează dacă se
                                          află în interval si se
                                          returnează în BC.
                LD    HL,+2D2B            Se face ca adresa de revenire
                PUSH  HL                  să fie cea a subrutinei
                                          STACK-BC.
                PUSH  BC                  Se face un salt indirect la
                RET                       locatia cerută.
```

Notă: Este interesant faptul că registrul pereche IY este reinitializat cînd s-a făcut revenirea din STACK-BC, dar 'H'L', care contine indicatorul următorului literal, nu este restaurat si ar putea fi conturbat. Pentru o revenire corectă în BASIC, H'L' trebuie să iasă din codul mașină continînd adresa SCANNING din instructiunea 'end-calc', 2758 hex (10072 zecimal).

THE 'USR STRING' FUNCTION (FUNCTIA 'USR SIR')
(Deplasament 19: 'usr-$')

Această subrutină tratează functia USR X$, unde X$ este un sir. Subrutina revine cu adresa bitului sablon în BC pentru corespondenta grafică definită de utilizator pentru X$. Ea dă prezentarea A dacă X$ nu este o singură literă între a si u sau un singur semn grafic definit de utilizator.

```
34BC usr-$      CALL  2BF1,STK-FETCH      Se aduc parametrii sirului
                                          X$.
                DEC   BC                  Se decrementează lungimea cu
                                          1 pentru a o testa.
                LD    A,B                 Dacă lungimea a fost diferită
                OR    C                   de 1, atunci se sare pentru a
                JR    NZ,34E7,REPORT-A    da prezentarea A.
                LD    A,(DE)              Se aduce codul singular al
                                          sirului.
                CALL  2C8D,ALPHA          Denotă acesta o literă?
                JR    C,34D3,USR-RANGE    Dacă da, salt pentru aducerea
                                          adresei sale.
                SUB   +90                 Se reduce intervalul pentru
                                          graficele actuale definite de
                                          utilizator la 0 - 20 zecimal.
                JR    C,34E7,REPORT-A     Se dă prezentarea A dacă se
                                          iasă din interval.
                CP    +15                 Se testează din nou
                                          intervalul.
                JR    NC,34E7,REPORT-A    Se dă prezentarea A dacă este
                                          afară din interval.
                INC   A                   Se face intervalul graficelor
                                          definite de utilizator 1 la
                                          21 zecimal, ca pentru a la u.
34D3 USR-RANGE  DEC   A                   Acum se face intervalul 0 la
                                          20 zecimal în fiecare caz.
                ADD   A,A                 Se înmulteste cu 8 pentru a
                ADD   A,A                 da un deplasament pentru
                ADD   A,A                 adresă.
                CP    +A8                 Se testează intervalul pentru
                                          deplasament.
                JR    NC,34E7,REPORT-A    Se dă prezentarea A dacă este
                                          în afara intervalului.
                LD    BC,(UDG)            Se aduce adresa primului
                                          primul grafic definit de
                                          utilizator în BC.
                ADD   A,C                 Se adună C la deplasament.
                LD    C,A                 Se stochează din nou
                                          rezultatul în C.
                JR    NC,34E4,USR-STACK   Salt dacă nu este nici un
                                          transport.
                INC   B                   Se incrementează B pentru a
                                          completa adresa.
                JP    2D2B,STACK-BC       Salt pentru a stoca adresa.
```

Prezentarea A - Argument invalid

```
34E7 REPORT-A   RST   0008,ERROR-1        Se apelează rutina de tratare
                DEFB  +09                 a erorii.
```

THE 'TEST-ZERO' SUBROUTINE (SUBRUTINA 'TEST ZERO')
Această subrutină este apelată de cel putin nouă ori pentru a testa dacă un număr în virgulă mobilă este zero. Acest test necesită ca primii patru octeti ai numărului să fie fiecare zero. Subrutina revine cu fanionul de trasport setat dacă numărul a fost de fapt zero.

```
34E9 TEST-ZERO   PUSH  HL          Se salvează HL în stivă.
                 PUSH  BC          Se salvează BC în stivă.
                 LD    B,A         Se salvează valoarea din A în
                                   B.
                 LD    A,(HL)      Se aduce primul octet.
                 INC   HL          Se indică al doilea octet.
                 OR    (HL)        Se face SAU primul octet cu
                                   al doilea.
                 INC   HL          Se indică al treilea octet.
                 OR    (HL)        Se face SAU rezultatul cu al
                                   treilea octet.
                 INC   HL          Se indică al patrulea octet.
                 OR    (HL)        Se face SAU rezultatul cu al
                                   patrulea octet.
                 LD    A,B         Se readuce valoarea initială
                                   a lui A.
                 POP   BC          Si a lui BC.
                 POP   HL          Se readuce indicatorul
                                   numărului în HL.
                 RET   NZ          Se revine cu transportul
                                   resetat dacă oricare din cei
                                   patru octeti a fost diferit
                                   de zero.
                 SCF               Se setează fanionul de
                 RET               transport pentru a indica
                                   faptul că acel număr a fost
                                   zero, si se revine.
```

THE 'GREATER THAN ZERO' OPERATION (OPERATI 'MAI MARE CA ZERO')
(Deplasament 37: 'greater-0')

Această subrutină redă o 'ultimă valoare' unu dacă prezenta 'ultimă valoare' este mai mare decît zero si dă zero în caz contrar. Ea este de asemenea folosită de către alte subrutine pentru a executa 'salt la plus'.

```
34F9 GREATER-0   CALL  34E9,TEST-ZERO   Este 'ultima valoare' zero?
                 RET   C               Dacă da, revenire.
                 LD    A,+FF           Salt mai departe la LESS THAN
                 JR    3507,IGN-TO-C   ZERO ???
```

THE 'NOT' FUNCTION (FUNCTIA 'NOT')
(Deplasament 30: 'not')

Această subrutină redă o 'ultimă valoare' unu dacă prezenta 'ultimă valoare' este zero si zero în caz contrar. Ea este de asemenea folosită de alte subrutine pentru a executa 'salt la zero'.

```
3501 NOT         CALL  34E9,TEST-ZERO   Fanionul de transport va fi
                                        setat numai dacă 'ultima
                                        valoare' este zero; aceasta
                                        dă rezultatul corect.
                 JR    3508,FP-0/1      Salt înainte.
```

THE 'LESS THAN ZERO' OPERATION (OPERATIA 'MAI MIC CA ZERO')
(Deplasament 36: 'less-0')

Această subrutină redă o 'ultimă valoare' unu dacă prezenta 'ultimă valoare' este mai mică decît zero si zero în caz contrar. Ea este de asemenea folosită si de alte subrutine pentru a executa 'salt la minus'.

```
3506 less-0      XOR   A             Se sterge registrul A.
3507 SIGN-TO-C   INC   HL            Se indică octetul de semn.
                 XOR   (HL)          Transportul este resetat
                 DEC   HL            pentru un număr pozitiv si
                 RLCA                este setat pentru un număr
                                     negativ; cînd se intră din
                                     GREATER-0 semnul opus trece
                                     în transport.
```

THE 'ZERO OR ONE' SUBROUTINE (SUBRUTINA 'ZERO SAU UNU')
Această subrutină setează 'ultima valoare' pe zero dacă fanionul de transport este resetat si o setează pe unu dacă fanionul de transport este setat. Cînd

este apelată din 'E-TO-FP' oricum creează zero sau unu, nu în stivă ci în memo.

```
3508 FP-0/1     PUSH  HL          Se salvează indicatorul
                                  rezultatului.
                LD    A,+00       Se sterge A fără a afecta
                                  transportul.
                LD    (HL),A      Se setează primul octet pe
                                  zero.
                INC   HL          Se indică al doilea octet.
                LD    (HL),A      Se setează al doilea octet
                                  pe zero.
                INC   HL          Se indică al treilea octet.
                RLA               Se roteste transportul în A,
                                  se face A unu dacă
                                  transportul a fost setat si
                                  zero dacă transportul a fost
                                  resetat.
                LD    (HL),A      Se setează al treilea octet
                                  pe unu sau pe zero.
                RRA               Asigurare că A este zero din
                                  nou.
                INC   HL          Se indică al patrulea octet.
                LD    (HL),A      Se setează al patrulea octet
                                  pe zero.
                INC   HL          Se indică al cincilea octet.
                LD    (HL),A      Se setează al cincilea octet
                                  pe zero.
                POP   HL          Se readuce indicatorul
                                  rezultat.
                RET               Sfîrsit.
```

THE 'OR' OPERATION (OPERATIA 'OR')
(Deplasament 07: 'or')

Această subrutină execută operatia binară 'X OR Y' ('X SAU Y') si redă X dacă Y este zero si valoarea zero în caz contrar.

```
3518 or         EX    DE,HL       HL indică Y, cel de al doilea
                                  număr.
                CALL  34E9,TEST-ZERO  Se testează dacă Y este zero.
                EX    DE,HL       Se refac indicatorii.
                RET   C           Se revine dacă Y a fost zero;
                                  acum X este 'ultima valoare'.
                SCF               Se setează fanionul de
                JR    3508,FP-0/1 transport si salt înapoi
                                  pentru a seta 'ultima
                                  valoare' pe 1.
```

THE 'NUMBER AND NUMBER' OPERATION (OPERATIA 'NUMAR SI NUMAR')
(Deplasament 08: 'no-&-no')

Această subrutină execută operatia binară 'X AND Y' ('X SI Y') si redă X dacă Y este diferit de zero si valoarea zero în caz contrar.

```
3524 no-&-no    EX    DE-HL       HL îl indică pe Y, DE pe X.
                CALL  34E9,TEST-ZERO  Se testează dacă Y este zero.
                EX    DE,HL       Se reschimbă indicatorii.
                RET   NC          Se revine cu X ca 'ultima
                                  valoare' dacă Y a fost
                                  diferit de zero.
                AND   A           Se reseează fanionul de
                JR    3508,FP-0/1 transport si salt înapoi
                                  pentru a seta 'ultima
                                  valoare' pe zero.
```

THE 'STRING AND NUMBER' OPERATION (OPERATIA 'SIR SI NUMAR')
(Deplasament 10: 'str-&-no')

Această subrutină execută operatia binară 'X$ AND Y' ('X$ SI Y') si redă X$ dacă Y este diferit de zero si un sir nul în caz contrar.

```
3528 str-&-no   EX    DE,HL       HL îl indică pe Y, DE pe X$.
                CALL  34E9,TEST-ZERO  Se testează dacă Y este zero.
                EX    DE,HL       Se schimbă indicatorii la loc
                RET   NC          Se revine cu X$ ca 'ultima
                                  valoare' dacă a fost diferit
                                  de zero.
                PUSH  DE          Se salvează indicatorul
```

```
                    DEC   DE                  numàrului.
                                              Se indicà al cincilea octet
                                              al parametrului sir, adicà
                                              octetul cel mai semnificativ
                                              al lungimii.
                    XOR   A                   Se sterge registrul A.
                    LD    (DE),A              Octetul cel mai semnificativ
                                              al lungimii este acum setat
                                              pe zero.
                    DEC   DE                  Se indicà octetul cel mai
                                              putin semnificativ al
                                              lungimii.
                    LD    (DE),A              Acum octetul cel mai putin
                                              semnificativ al lungimii este
                                              setat pe zero.
                    POP   DE                  Se readuce indicatorul.
                    RET                       Revenire cu parametrul sir
                                              ca fiind 'ultima valoare'.
```

THE 'COMPARISON' OPERATIONS (OPERATIA 'COMPARARE')
(Deplasamente 09 la 0E & 11 la 16: 'no-l-eql', 'no-gr-eq', 'nos-neql', 'no-grtr', 'no-less', 'nos-eql', 'str-l-eql', 'str-gr-eq', strs-neql', 'str-grtr',str-less' &strs-eql')

Această subrutină este folosită pentru a executa 12 operatii de comparare posibile. Deplasamentul operatiei singulare este prezent în registrul B la începutul acestei subrutine.

```
353B no-l-eql       LD    A,B                 Deplasamentul singular trece
     etc.                                     în registrul A.
                    SUB   +08                 Intervalul este acum 01-06 &
                                              09-0E.
                    BIT   2,A                 Acest interval este schimbat
                    JR    NZ,3543,EX-OR-NOT   cu: 00-02, 04-06, 08-0A &
                    DEC   A                   0C-0E.
3543 EX-OR-NOT      RRCA                      Apoi este redus la 00-07 cu
                                              transportul setat pentru
                                              'mai mare sau egal' & 'mai
                                              mic'; apoi operatiile cu
                    JR    NC,354E,NU-OR-STR   transportul setat sînt
                    PUSH  AF                  tratate ca si operatiile de
                    PUSH  HL                  complementare fatà de unu ce
                    CALL  343C,EXCHANGE       schimbà valoarea.
                    POP   DE
                    EX    DE,HL
                    POP   AF
354E,NU-OR-STR      BIT   2,A                 Comparatiile numerice sînt
                    JR    NZ,3559,STRINGS     acum separate de compartiile
                                              de siruri, testînd bitul 2.
                    RRCA                      Operatiile numrice au acum
                                              intervalul 00-01 cu
                                              transportul setat pentru
                                              'egal' si 'diferit'.
                    PUSH  AF                  Se salveazà deplasamentul.
                    CALL  300F,SUBTRACT       Numerele sînt scàzute pentru
                    JR    358C,END-TESTS      testul final.
3559 STRINGS        RRCA                      Comparatiile de siruri au
                                              acum intervalul 02-03 cu
                                              transportul setat pentru
                                              'egal' si 'diferit'.
                    PUSH  AF                  Salvare deplasament.
                    CALL  2BF1,STK-FETCH      Se aduc acum din stiva
                    PUSH  DE                  calculatorului lungimile si
                    PUSH  BC                  adresele de bazà ale
                    CALL  2BF1,STK-FETCH      sirurilor.
                    POP   HL                  Lungimea celui de al doilea
                                              sir.
3564 BYTE-COMP      LD    A,H
                    OR    L
                    EX    (SP),HL
                    LD    A,B
                    JR    NZ,3575,SEC-PLUS    Salt numai dacà al doilea sir
                    OR    C                   este nul.
356B SECND-LOW      POP   BC                  Aici al doilea sir este fie
                                              nul fie mai mic decît primul.
                    JR    Z,3572,BOTH-NULL
                    POP   AF
                    CCF                       Transportul este complementat
                    JR    3588,STR-TEST       pentru a da corect testul
```

```
                                                    rezultatului.
3572 BOTH-NULL      POP   AF                        Aici transportul este
                    JR    3588,STR-TEST             utilizat cum s-a mentinut.
3575 SEC-PLUS       OR    C
                    JR    Z,3585,FRST-LESS          Acum primul sir este nul, al
                                                    doilea sir nu este nul.
                    LD    A,(DE)                    Nici un sir nu este nul, asa
                    SUB   (HL)                      cà se compará urmàtorii lor
                                                    octeti.
                    JR    C,3585,FRST-LESS          Primul octet este mai mic.
                    JR    NZ,356B,SECND-LOW         Al doilea octet este mai mic.
                    DEC   BC                        Octetii sînt egali; asa cà
                    INC   DE                        lungimile sînt decrementate
                    INC   HL                        si se face un salt la BYTE-
                    EX    (SP),HL                   COMP pentru a compara
                    DEC   HL                        urmàtorii octeti ai
                                                    sirurilor reduse.
                    JR    3564,BYTE-COMP
3585 FRST-LESS      POP   BC
                    POP   AF
                    AND   A                         Aici transportul este sters
                                                    pentru un test corect
                                                    rezultat.
3588 STR-TEST       PUSH  AF                        Pentru testele sirului, se
                    RST   0028,FP-CALC              pune un zero în stiva
                                                    calculatorului.
                    DEFB  +A0,stk-zero
                    DEFB  +38,end-calc
358C END-TESTS      POP   AF                        Aceste trei teste, apelate
                    PUSH  AF                        dupà cum este necesar, dau
                    CALL  C,3501,NOT                rezultatele corecte pentru
                    POP   AF                        toate cele 12 comparatii.
                    PUSH  AF                        Transportul initial este
                    CALL  NC,34F9,GREATER-0         setat pentru 'diferit' si
                    POP   AF                        'egal', iar transportul final
                                                    este pentru 'mai mare', 'mai
                                                    mic' si 'egal'.
                    RRCA
                    CALL  NC,3501,NOT
                    RET                             Sfîrsit.
```

THE 'STRING CONCATENATION' OPERATION (OPERATIA 'CONCATENARE SIR')
(Deplasament 17: 'strs-add')

Aceastà subrutinà executà operatia binarà 'A$+B$'. Parametrii acestor siruri sînt adusi si este gàsità lungimea totalà. Se face accesibil suficient spatiu în spatiul de lucru pentru a contine ambele siruri si se copiazà sirurile. Rezultatul acestei subrutine este de aceea sà producà o variabila temporarà A$+B$ care este rezidentà în spatiul de lucru.

```
359C strs-add       CALL  2BF1,STK-FETCH            Se aduc si se salveazà
                    PUSH  DE                        parametrii celui de al doilea
                                                    sir.
                    PUSH  BC
                    CALL  2BF1,STK-FETCH            Se aduc parametrii primului
                                                    sir.
                    POP   HL
                    PUSH  HL                        Acum lungimile sînt în Hl si
                                                    în DE.
                    PUSH  DE                        Se salveazà parametrii
                    PUSH  BC                        primului sir.
                    ADD   HL,BC                     Se calculeazà lungimea totalà
                    LD    B,H                       a celor douà siruri si se
                    LD    C,L                       trece în BC.
                    RST   0030,BC-SPACES            Se face accesibil suficient
                                                    spatiu.
                    CALL  2AB2,STK-STORE            Parametrii noului sir sînt
                                                    trecuti în stiva
                                                    calculatorului.
                    POP   BC                        Parametrii primului sir sînt
                    POP   HL                        regàsiti si sirul este copiat
                    LD    A,B                       în spatiu de lucru atîta timp
                    OR    C                         cît nu este un sir nul.
                    JR    Z,35B7,OTHER-STR
                    LDIR
35B7 OTHER-STR      POP   BC                        Exact aceeasi procedurà este
                    POP   HL                        urmatà de al doilea sir
                    LD    A,B                       rezultînd astfel 'A$+B$'.
                    OR    C
```

```
                JR    Z,35BF,STK-PNTRS
                LDIR
```

THE 'STK-PNTRS' SUBROUTINE (SUBRUTINA 'STK-PNTRS')
Această subrutină resetează registrul pereche HL astfel încît să indice primul octet al 'ultimei valori', adică STKEND-5, iar registrul pereche DE să indice cu unu după 'ultima valoare', adică STKEND.

```
35BF STK-PNTRS  LD    HL,(STKEND)       Se aduce valoarea curentă a
                                        lui STKEND.
                LD    DE,+FFFB          Se setează DE pe -5,
                                        complemetînd față de doi.
                PUSH  HL                Se stochează valoarea lui
                                        STKEND.
                ADD   HL,DE             Se calculează STKEND-5.
                POP   DE                Acum DE contine STKEND si HL
                RET                     contine STKEND-5.
```

THE 'CHR$' FUNCTION (FUNCTIA 'CHR$')
(Deplasament 2F: 'chr$')

Această subrutină tratează functia CHR$ X si creează un sir caracter singular în spatiul de lucru.

```
35C9 chr$       CALL  2DD5,FP-TO-A      'Ultima valoare' este
                                        comprimată în registrul A.
                JR    C,35DC,REPORT-B   Se prezintă eroare dacă X a
                JR    NZ,35DC,REPORT-B  fost mai mare decît 255
                                        zecimal, sau X a fost un
                                        număr negativ.
                PUSH  AF                Se salvează valoarea
                                        comprimată a lui X.
                LD    BC,+0001          Se face accesibil un spatiu
                RST   0030,BC-SPACES    în spatiul de lucru.
                POP   AF                Se aduce valoarea.
                LD    (DE),A            Se copiază valoarea în
                                        spatiul de lucru.
                CALL  2AB2,STK-STORE    Se trec parametrii noului sir
                                        în stiva calculatorului.
                EX    DE,HL             Se resetează indicatorii.
                RET                     Sfîrsit.
```

Prezentarea B - Intreg afară din interval.

```
35DC REPORT-B   RST   0008,ERROR-1      Se apelează rutina de tratare
                DEFB  +0A               eroare.
```

THE 'VAL' AND 'VAL$' FUNCTION (FUNCTIA 'VAL' SI 'VAL$')
(Deplasamente 1D: 'val' si 18: 'val$')

Această subrutină tratează functiile VAL X$ si VAL$ X$. Cînd tratează VAL X$, ea revine cu o 'ultimă valoare' care este rezultatul evaluării sirului (fără cotele limită) ca o expresie numerică. Cînd tratează VAL$ X$, ea evaluează X$ (fără cotele limită) ca o expresie sir, si returnează parametrii acestei expresii sir ca o 'ultimă valoare' în stiva calculatorului.

```
35DE val        LD    HL,(CH-ADD)       Valoarea curentă a lui CH-ADD
    (also val$) PUSH  HL                este păstrată în stiva
                                        masinii.
                LD    A,B               'Deplasamentul' lui 'val' sau
                                        a lui 'val$' trebuie să fie
                                        în registrul B; acum el este
                                        copiat în A.
                ADD   A,+E3             Se produce +00 si transport
                                        setat pentru 'val', +FB si
                                        transport resetat pentru
                                        'val$'.
                SBC   A,A               Este produs +FF (bitul 6 este
                                        astfel setat) pentru 'val',
                                        si +00 (bitul 6 este resetat)
                                        pentru 'val$'.
                PUSH  AF                Se salvează 'fanionul' în
                                        stiva masinii.
                CALL  2BF1,STK-FETCH    Se aduc parametrii sirului;
                PUSH  DE                salvare adresa de început;
                INC   BC                se adună un octet la lungime
                RST   0030,BC-SPACES    si se face accesibil spatiu
                                        pentru sir (+1) în spatiul de
                                        lucru.
```

```
                POP   HL                  Adresa de început a sirului
                                          trece în HL ca adresă sursă.
                LD    (CH-ADD),DE         Indicatorul primului spatiu
                PUSH  DE                  nou trece în CH-ADD si în
                                          stiva masinii.
                LDIR                      Sirul este copiat în spatiul
                                          de lucru, împreună cu un
                                          octet aditional.
                EX    DE,HL               Se schimbă indicatorii.
                DEC   HL                  Octetul aditional este
                LD    (HL),+0D            înlocuit cu un caracter
                                          'carriage return'.
                RES   7,(FLAGS)           Fanionul de sintaxă este
                CALL  24FB,SCANNING       resetat si sirul este baleiat
                                          pentru sintaxă corectă.
                RST   0018,GET-CHAR       Se aduce caracterul de după
                                          sir.
                CP    +0D                 Se testează dacă s-a ajuns la
                                          sfîrsitul expresiei.
                JR    NZ,360C,V-RPORT-C   Dacă nu, se prezintă eroare.
                POP   HL                  Se aduce adresa de început a
                                          sirului.
                POP   AF                  Este adus 'fanionul' pentru
                XOR   (FLAGS)             'val/val$' si bitul 6 este
                AND   +40                 comparat cu bitul 6 al
                                          rezultatului baleierii
                                          sintaxei.
360C V-RPORT-C  JP    NZ,1C8A,REPORT-C    Se prezintă eroare dacă cei
                                          doi biti nu coincid.
                LD    (CH-ADD),HL         Din nou adresa de început în
                                          CH-ADD.
                SET   7,(FLAGS)           Fanionul este setat pentru
                                          executie linie.
                CALL  24FB,SCANNING       Sirul este tratat ca o
                                          'expresie următoare' si este
                                          produsă o 'ultimă valoare'.
                POP   HL                  Se readuce valoarea initială
                LD    (CH-ADD),HL         a lui CH-ADD.
                JR    35BF,STK-PNTRS      Subrutina iese prin STK-PNTRS
                                          care resetează indicatorii.
```

THE 'STR$' FUNCTION (FUNCTIA 'STR$')
(Deplasament 2E: 'str$')

Această subrutină tratează functia STR$ X si returnează o 'ultimă valoare' care este un set de parametrii care definesc un sir continut care ar apărea pe ecran dacă X ar fi afisat de o comandă PRINT.

```
361F str$       LD    BC,+0001            Se face un spatiu în spatiul
                RST   0030,BC-SPACES      de lucru si adresa lui este
                LD    (K-CUR),HL          copiată în K-CUR, adresa
                                          cursorului.
                PUSH  HL                  Această adresă este salvată
                                          de asemenea în stivă.
                LD    HL,(CURCHL)         Adresa canalului curent este
                PUSH  HL                  salvată în stiva masinii.
                LD    A,+FF               Este deschis canalul 'R',
                CALL  1601,CHAN-OPEN      permitînd 'tipărirea' sirului
                                          afară din spatiul de lucru.
                CALL  2DE3,PRINT-FP       'Ultima valoare', X, este
                                          acum tipărită în afara
                                          spatiului de lucru si spatiul
                                          de lucru este extins cu
                                          fiecare caracter.
                POP   HL                  Se readuce CURCHL în HL si se
                CALL  1615,CHAN-FLAG      readuc fanioanele
                                          corespunzătoare.
                POP   DE                  Se readuce adresa de început
                                          a sirului.
                LD    HL,(K-CUR)          Acum adresa cursorului este
                AND   A                   una după sfîrsitul sirului si
                SBC   HL,DE               deci diferenta este lungimea.
                LD    B,H                 Se transferă lungimea în BC.
                LD    C,L
                CALL  2AB2,STK-STO-$      Se trec în stiva
                                          calculatorului parametrii
                                          noului sir.
                EX    DE,HL               Se resetează indicatorii.
                RET                       Sfîrsit.
```

Notă: Vezi PRINT-FP pentru lămurirea erorii 'PRINT "A"+STR$ 0.1'.

THE 'READ-IN' SUBROUTINE (SUBRUTINA 'READ-IN')
(Deplasament 1A: 'read-in')

Subrutina este apelată prin intermediul deplasamentului calculatorului în cadrul primei linii a rutinei S-INKEY$ în SCANNING. Aceasta vine să furnizeze informatiile citite de-a lungul diferitelor siruri accesibile în standardul Spectrum. Ca si INKEY$, subrutina revine cu un sir.

```
3645 read-in      CALL  1E94                  Parametrul numeric este
                                              comprimat în registrul A.
                  CP    +10                   Este mai mic decît 16
                                              zecimal?
                  JP    NC,1E9F,REPORT-B      Dacă nu, se prezintă eroare.
                  LD    HL,(CURCHL)           Adresa canalului curent este
                  PUSH  HL                    salvată în stiva masinii.
                  CALL  1601,CHAN-OPEN        Se deschide canalul care este
                                              specificat de către
                                              parametru.
                  CALL  15E6,INPUT-AD         Acum semnalul este acceptat,
                                              ca o 'valoare-tastă'.
                  LD    BC,+0000              Lungimea absentă a sirului
                                              rezultant este zero.
                  JR    NC,365F,R-I-STORE     Salt dacă nu este nici un
                                              semnal.
                  INC   C                     Acum se setează lungimea pe
                                              1.
                  RST   0030,BC-SPACES        Se face un spatiu în spatiul
                                              de lucru.
                  LD    (DE),A                Se pune sirul în acest spatiu
365F R-I-STORE    CALL  2AB2,STK-STO-$        Se trec parametrii sirului în
                                              stiva calculatorului.
                  POP   HL                    Se readuce CURCHL si
                  CALL  1615,CHAN-FLAG        fanioanele corespunzătoare.
                  JP    35BF,STK-PNTRS        Iesire, setînd indicatorii.
```

THE 'CODE' FUNCTION (FUNCTIA 'CODE')
(Deplasament 1C: 'code')

Această subrutină tratează functia CODE A$ si revine codul Spectrum al primului caracter în A$, sau zero dacă A$ trebuie să fie nul.

```
3669 code         CALL  2BF1,STK-FETCH        Se aduc parametrii sirului.
                  LD    A,B                   Este testată lungimea si
                  OR    C                     registrul A continînd zero
                  JR    Z,3671,STK-CODE       este transportat mai departe
                                              ca si cînd A$ este u sir nul.
                  LD    A,(DE)                Altfel codul primului
                                              caracter este pus în A.
3671 STK-CODE     JP    2D28,STACK-A          Această subrutină iese prin
                                              STACK-A care dă 'ultima
                                              valoare' corectă.
```

THE 'LEN' FUNCTION (FUNCTIA 'LEN')
(Deplasament 1E: 'len')

Această subrutină tratează functia LEN A$ si returnează o 'ultimă valoare' care este egală cu lungimea sirului.

```
3674 len          CALL  2BF1,STK-FETCH        Sînt adusi parametrii sirului
                  JP    2D2B,STACK-B          Subrutina iese prin STACK-BC,
                                              care dă 'ultima valoare'
                                              corectă.
```

THE 'DECREASE THE COUNTER' SUBROUTINE (SUBRUTINA 'DECREMENTARE CONTOR')
(Deplasament 35: 'dec-jr-nz')

Această subrutină este apelată doar de subrutina SERIES GENERATOR si de fapt este o operatie 'DJNZ' dar contorul este varibila sistem BREG, mai degarbă decît registrul B.

```
367A dec-jr-nz    EXX                         Se trece la setul registrilor
                  PUSH  HL                    auxiliari si se salvează
                                              indicatorul următorului
                                              literal în stiva masinii.
                  LD    HL,+5C67              HL va indica BREG.
```

```
                DEC   (HL)                  Se decrementează BREG.
                POP   HL                    Se readuce indicatorul
                                            următorului literal.
                JR    NZ,3687,JUMP-2        Se execută salt dacă este
                                            diferit de zero.
                INC   HL                    Se trece peste următorul
                                            literal.
                EXX                         Se revine la setul principal
                                            de registrii.
                RET                         Sfîrsit.
```

THE 'JUMP' SUBROUTINE (SUBRUTINA 'JUMP' ('SALT'))
(Deplasament 33: 'jump')

Subrutina execută un salt neconditionat cînd este apelată prin literalul '33'. De asemenea ea mai este folosită de către subrutinele DECREASE THE COUNTER si JUMP ON TRUE.

```
3686 JUMP       EXX                         Se trece în setul registrilor
                                            auxiliari.
3687 JUMP-2     LD    E,(HL)                Următorul literal (salt
                                            lungime) este pus în
                                            registeul E.
                LD    A,E                   Numărul 00 hex sau FF hex
                RLA                         este format în A în
                SBC   A,A                   concordantă cu faptul dacă E'
                LD    D,A                   este pozitiv sau negativ, si
                                            apoi este copiat în D'.
                ADD   HL,DE                 Acum registrii H' & L'
                EXX                         contin indicatorul
                                            următorului literal.
                RET                         Sfîrsit.
```

THE 'JUMP ON TRUE' SUBROUTINE (SUBRUTINA 'JUMP ON TRUE' ('SALT CONDITIONAT')
(Deplasament 00: 'jump-true')

Această subrutină execută un salt conditionat dacă 'ultima valoare' din stiva calculatorului, sau mai exact numărul curent adresat de registrul pereche DE, este adevărat.

```
368F jump-true  INC   DE                    Se indică al treilea octet,
                INC   DE                    care este zero sau unu.
                LD    A,(DE)                Se colectează acest octet în
                                            registrul A.
                DEC   DE                    Se indică încă o dată primul
                DEC   DE                    octet.
                AND   A                     Se testează al treilea octet:
                                            este zero?
                JR    NZ,3686,JUMP          Se execută saltul dacă
                                            octetul este diferit de zero,
                                            adică numărul nu este fals.
                EXX                         Se trece la setul registrilor
                                            auxiliari.
                INC   HL                    Se trece peste lungime salt.
                EXX                         Se revine la setul
                                            registrilor principali.
                RET                         Sfîrsit.
```

THE 'END-CALC' SUBROUTINE (SUBRUTINA 'END-CALC')
(Deplasament 38: 'end-calc')

Această subrutină terminâ o operatie RST 0028.

```
369B end-calc   POP   AF                    Este înlăturată adresa de
                                            revenire în calculator
                                            ('RE-ENTRY').
                EXX                         In schimb, adresa din H'L'
                EX    (SP),HL               este pusă în stiva masinii si
                EXX                         se face un salt indirect la
                                            ea. H'L' va contine acum
                                            orice adresă anterioară în
                                            lantul de adrese al
                                            calculatorului.
                RET                         Sfîrsit.
```

THE 'MODULUS' SUBROUTINE (SUBRUTINA 'MODUL')
(Deplasament 32: 'n-mod-m')

Această subrutină calculează N (mod M), unde N este un întreg pozitiv continut în vîrful stivei calculatorului, 'ultima valoare', iar N este un întreg continut în în stivă dedesubt de M.

Subrutina revine cu partea întreagă a cîtului INT (N/M) în vîrful stivei calculatorului, 'ultima valoare', iar restul N-INT (N/M) în locul al doilea din stivă.

Această subrutină este apelată de-a lungul calculului unui număr oarecare, pentru a reduce N mod 65537 zecimal.

```
36A0 n-mod-m    RST   0028,FP-CALC          N,M
                DEFB  +C0,st-mem-0          N,M      mem-0 contine M
                DEFB  +02,delete            N
                DEFB  +31,duplicate         N,N
                DEFB  +E0,get-mem-0         N,N,M
                DEFB  +05,division          N,N/M
                DEFB  +27,int               N,INT (N/M)
                DEFB  +E0,get-mem-0         N,INT (N/M),M
                DEFB  +01,exchange          N,M,INT (N/M)
                DEFB  +C0,st-mem-0          N,M,INT (N/M)  mem-0 contine
                                            INT (N/M)
                DEFB  +04,multiply          N,M*INT (N/M)
                DEFB  +03,subtract          n-M*INT (N/M)
                DEFB  +E0,get-mem-0         n-M*INT (N/M),INT (N/M)
                DEFB  +38,end-calc
                RET                         Sfîrsit.
```

THE 'INT' FUNCTION (FUNCTIA 'INT')
(Deplasament 27: 'int')

Această subrutină tratează functia INT X si redă o 'ultimă valoare' care este 'partea întreagă' a valorii înlocuite. Astfel INT 2.4 dă 2 dar cum subrutina întotdeauna rotunjeste rezultatul în jos, INT -2.4 va da -3.

Subrutina foloseste subrutina INTEGER TRUNCATION TOWARDS ZERO de la 3214 pentru a produce I (X) astfel încît I (2.4) dă 2 si I (-2.4) dă -3. Astfel, INT X este dat de I (X) pentru valorile lui X care sînt mai mari sau egale cu zero, si I (X)-1 pentru valorile nehative ale lui X care încă nu sînt întregi, cînd rezultatul este, bineînteles, I (X).

```
36AF int        RST   0028,FP-CALC          X
                DEFB  +31,duplicate         X,X
                DEFB  +36,less-0            X,(1/0)
                DEFB  +00,jump-true         X
                DEFB  +04,to 36B7,X-NEG     X
```

Pentru valori ale lui X care s-au arătat a fi mai mari sau egale cu zero nu se face nici un salt si I (X) este gata găsit.

```
                DEFB  +3A,truncate          I (X)
                DEFB  ++38,end-calc
                RET                         Sfîrsit.
```

Cînd X este un întreg negativ I (X) este returnat, altfel se returnează I (X)-1.

```
36B7 X-NEG      DEFB  +31,duplicate         X,X
                DEFB  +3A,truncate          X,I (X)
                DEFB  +C0,st-mem-0          X,I (X)   mem-0 contine I (X)
                DEFB  +03,subtract          X-I (X)
                DEFB  +E0,get-mem-0         X-I (X),I (X)
                DEFB  +01,exchange          I (X),X-I (X)
                DEFB  +30,not               I (X),(1/0)
                DEFB  +00,jump-true         I (X)
                DEFB  +03,to 36C2,EXIT      I (X)
```

Saltul este executat pentru valori ale lui X care sînt întregi negativi, altfel nu se face nici un salt si se calculează I (X)-1.

```
                DEFB  +A1,stk-one           I (X),1
                DEFB  +03,subtract          I (X)-1
```

In fiecare caz subrutina se termină cu:

```
36C2 EXIT       DEFB  +38,end-calc          I (X) sau I (X)-1
                RET
```

THE 'EXPONENTIAL' FUNCTION (FUNCTIA 'EXPONENTIAL')
(Deplasament 26: 'exp')

Această subrutină tratează functia EXP X si este prima dintre cele patru

rutine care foloseste SERIES GENERATOR pentru a produce polinoame Cebîsev.
Aproximarea pentru EXP X se gàseste în felul urmàtor:

i. X este împàrtit cu LN 2 pentru a da Y, asa cà 2 la puterea Y este acum rezultatul cerut.

ii. Se gàseste valoarea N, astfel încît N=INT Y.

iii. Se gàseste valoarea lui W, astfel încît W=Y-N, unde 0<=W<=1, asa cum este necesar pentru ca seriile sà fie convergente.

iv. Este format argumentul Z, astfel încît Z=2*W-1.

v. SERIES GENERATOR este folosit pentru a reda 2**W.

vi. In final N este adunat la exponent, rezultînd 2**(N+W), care este 2**Y si de aceearàspunsul cerut pentru EXP X.

Metoda folositä este ilustratà folosind un program BASIC în Anexà.

```
36C4 EXP        RST   0028,FP-CALC          X
```

Se executà pasul i.

```
                DEFB  +3D,re-stack          X (în forma în virgulà mobilà
                                               desfàsuratà)
                DEFB  +34,stk-data          X,1/LN 2
                DEFB  +F1,exponent+81
                DEFB  +38,+AA,+3B,+29
                DEFB  +04,multiply          X/LN 2 = Y
```

Se executà pasul ii.

```
                DEFB  +31,duplicate         Y,Y
                DEFB  +27,int,1C46          Y,INT Y = N
                DEFB  +C3-st-mem-3          Y,N          mem-3 contine N.
```

Se executà pasul iii.

```
                DEFB  +03,subtract          Y-N = W
```

Se executà pasul iv.

```
                DEFB  +31,duplicate         W,W
                DEFB  +0F,addition          2*W
                DEFB  +A1,stk-one           2*W,1
                DEFB  +03,subtract          2*W-1 = Z
```

Se executà pasul v, trecînd la SERIES GENERATOR parametrul '8' si cele opt constante cerute.

```
                DEFB  +88,series-08         Z
          1.    DEFB  +13,exponent+63
                DEFB  +36,(+00,00,+00)
          2.    DEFB  +58,exponent+68
                DEFB  +65,+66,(+00,+00)
          3.    DEFB  +9D,exponent+6D
                DEFB  +78,+65,+40,(+00)
          4.    DEFB  +A2,exponent+72
                DEFB  +60,+32,+C9,(+00)
          5.    DEFB  +E7,exponent+77
                DEFB  +21,+F7,+AF,+24
          6.    DEFB  +EB,exponent+7B
                DEFB  +2F,+B0,+B0,+14
          7.    DEFB  +EE,exponent+7E
                DEFB  +7E,+BB,+94,+58
          8.    DEFB  +F1,exponent+81
                DEFB  +3A,+7E,+F8,+CF
```

La sfîrsitul ultimei bucle 'ultima valoare' este 2**W.

Se executà pasul vi.

```
                DEFB  +E3,get-mem-3         2**W,N
                DEFB  +38,end-calc
                CALL  2DD5,FP-TO-A          Valoarea absolutà a lui N mod
                                            256 zecimal este pusà în
                                            registrul A.
                JR    NZ,3705,N-NEGTV       Salt înainte dacà N a fost
```

```
                        JR    C,3703,REPORT-6     negativ.
                                                  Este eroare dacà ABS N este
                                                  mai mare decît 255 zecimal.
                        ADD   A,(HL)              Acum se adunà ABS N la
                                                  exponent.
                        JR    NC,370C,RESULT-OK   Salt numai dacà e este mai
                                                  mare decît 255 zecimal.

Prezentarea 6 - Numàr prea mare

3703 REPORT-6           RST   0008,ERROR-1        Se apeleazà rutina de tratare
                        DEFB  +05                 eroare.

3705 N-NEGTV            JR    C,370E,RSLT-ZERO    Rezultatul trebuie sà fie
                                                  zero dacà N este mai mic
                                                  decît -255 zecimal.
                        SUB   (HL)                Se scade ABS N din exponent
                                                  ca si cum N ar fi fost
                                                  negativ.
                        JR    NC,370E,RSLT-ZERO   Rezultat zero dacà e este mai
                                                  mic decît zero.
                        NEG                       Minus e este schimbat în e.
370C RESULT-OK          LD    (HL),A              Se introduce exponentul, e.
                        RET                       Sfîrsit: 'ultima valoare'
                                                  este EXP X.
370E RSLT-ZERO          RST   0028,FP-CALC        Se foloseste calculatorul
                        DEFB  +02,delete          pentru a face 'ultima
                                                  valoare' zero.
                        DEFB  +A0,stk-zero
                        DEFB  +38,end-calc
                        RET                       Sfîrsit, cu EXP X = 0.
```

THE 'NATURAL LOGARITHM' FUNCTION (FUNCTIA 'LOGARITM NATURAL')
(Deplasament 25: 'ln')

Aceastà subrutinà trateazà functia LN X si este a doua din cele patru rutine care utilizeazà SERIES GENERATOR pentru a produce polinoamele Cebîsev.
Aproximarea lui N X se gàseste în modul urmàtor:

i. Se testeazà X si se dà prezentarea A dacà X nu este pozitiv.

ii. Apoi X este despàrtit în exponentul sàu adevàrat, e', si în mantisa sa X' = X/(2**e'), unde X' este mai mare sau egal cu 1.

iii. Se formeazà valoarea cerutà Y1 sau Y2. Dacà X' este mai mare decît 0.8 atunci Y1=e'*LN 2 si în caz contrar Y2=(e'-1)*LN 2.

iv. Dacà X' este mai mare decît 0.8 atunci cantitatea X'-1 este stocatà; altfel se stocheazà 2*X'-1.

v. Acum este format argumentul Z, care, dacà X' este mai mare decît 0.8, este Z=2.5*X'-3; altfel Z=5*X'-3. In fiecare caz, -1<=Z<=1, asa cum este necesar pentru ca seriile sà fie convergente.

vi. SERIES GENERATOR este folosit pentru a produce functia cerutà.

vii. In final o simplà înmultire si o adunare conduc la returnarea lui LN X ca 'ultima valoare'.

```
3713 ln                 RST   0028,FP-CALC       X

Se executà pasul i.

                        DEFB  +3D,re-stack        X (în forma în virgulà mobilà
                                                     desfàsuratà)
                        DEFB  +31,duplicate       X,X
                        DEFB  +37,greater-0       X,(1/0)
                        DEFB  +00,jump-true       X
                        DEFB  +04,to 371C,VALID   X
                        DEFB  ++38,end-calc       X

Prezentarea A - Argument invalid

371A REPORT-A           RST   0008,ERROR-1        Se apeleazà rutina de tratare
                        DEFB  +09                 eroare.

Se executà pasul ii.
```

```
371C VALID      DEFB  +A0,stk-zero          X,0              1 sters este
                DEFB  +02,delete           X      supraimprimat cu zero.
                DEFB  +38,end-calc         X
                LD    A,(HL)               Exponentul, e, trece în A.
                LD    (HL),+80             X este redus la X'.
                CALL  2D28,STACK-A         Stiva contine: X',e.
                RST   0028,FP-CALC         X',e
                DEFB  +34,stk-data         X',e,128 (zecimal)
                DEFB  +38,exponent+88
                DEFB  +00,(+00,+00,+00)
                DEFB  +03,subtract         X',e'
```

Se execută pasul iii.

```
                DEFB  ++01,exchange        e',X'
                DEFB  +31,duplicate        e',X',X'
                DEFB  +34,stk-data         e',X',X',0.8 (zecimal)
                DEFB  +F0,exponent+80
                DEFB  +4C,+CC,+CC,+CD
                DEFB  +03,subtract         e',X',X'-0.8
                DEFB  +37,greater-0        e',X',(1/0)
                DEFB  +00,jump-true        e',X'
                DEFB  +08,to 373D,GRE.8    e',X'
                DEFB  +01,exchange         X',e'
                DEFB  +A1,stk-one          X',e',1
                DEFB  +03,subtract         X',e'-1
                DEFB  +01,exchange         e'-1,X'
                DEFB  +38,end-calc         e'-1,X'
                INC   (HL)                 Se dubleazá X' pentru a da
                                           2*X'.
                RST   0028,FP-CALC         e'-1,2*X'
373D GRE.8      DEFB  +01,exchange         X',e             - X' mare
                                           2*X',e'-1        - X' mic
                DEFB  +34,stk-data         X',e',LN 2
                DEFB  +F0,exponent+80      2*X',e'-2,LN 2
                DEFB  +31,+72,+17,+F8
                DEFB  +04,multiply         X',e'*LN 2 = Y1
                                           2*X',(e'-1)*LN 2 = Y2
```

Se execută pasul iv.

```
                DEFB  +01,exchange         Y1,X'            - X' mare
                                           Y2,2*X'          - X' mic
                DEFB  +A2,stk-half         Y1,X',.5 (zecimal)
                                           Y2,2*X',.5
                DEFB  +03,subtract         Y1,X'-.5
                                           Y2,2*X'-.5
                DEFB  +A2,stk-half         Y1,X'-.5,.5
                                           Y2,2*X'-.5,.5
                DEFB  +03,subtract         Y1,X'-1
                                           Y2,2*X'-1
```

Se execută pasul v.

```
                DEFB  +31,duplicate        Y,X'-1,X'-1
                                           Y2,2*X'-1,2*X'-1
                DEFB  +34,stk-data         Y1,X'-1,X'-1,2.5 (zecimal)
                                           Y2,2*X'-1,2*X'-1,2.5
                DEFB  +32,exponent+82
                DEFB  +20,(+00,+00,+00)
                DEFB  +04,multiply         Y1,X'-1,2.5*X'-2.5
                                           Y2,2*X'-1,5*X'-2.5
                DEFB  +A2,stk-half         Y1,X'-1,2.5*X'-2.5,.5
                                           Y2,2*X'-1,5*X'-2.5,.5
                DEFB  +03,subtract         Y1,X'-1,2.5*X'-3=Z
                                           Y2,2*X'-1,5*X'-3=Z
```

Se execută pasul vi, trecînd în SERIES GENERATOR parametrul '12' zecimal, si a 12 constantă cerută.

```
                DEFB  +8C,series-0C        Y1,X'-1,Z sau Y2,2*X'-1,Z
          1.    DEFB  +11,exponent+61
                DEFB  +AC,(+00,+00,+00)
          2.    DEFB  +14,exponent+64
                DEFB  +09,(+00,+00,+00)
          3.    DEFB  +56,exponent+66
                DEFB  +DA,+A5,(+00,+00)
          4.    DEFB  +59,exponent+69
                DEFB  +30,+C5,(+00,+00)
          5.    DEFB  +5C,exponent+6C
```

```
                DEFB  +90,+AA,(+00,+00)
6.              DEFB  +9E,exponent+6E
                DEFB  +70,+6F,+61,+(+00)
7.              DEFB  +A1,exponent+71
                DEFB  +CB,+DA,+96,,(+00)
8.              DEFB  +A4,exponent+74
                DEFB  +31,+9F,+B4,(+00)
9.              DEFB  +E7,exponent+77
                DEFB  +A0,+FE,+5C,+FC
10.             DEFB  +EA,exponent+7A
                DEFB  +1B,+43,+CA,+36
11.             DEFB  +ED,exponent+7D
                DEFB  +A7,+9C,+7E,+5E
12.             DEFB  +F0,exponent+80
                DEFB  +6E,+23,+80,+93
```

La sfîrsitul ultimei bucle 'ultima valoare' este:

```
sau   LD X'/(X'-1)          pentru valorile mai mari
                            ale lui X'
sau   LD (2*X')/(2*X'-1)    pentru valorile mai mici
                            ale lui X'
```

Se executá pasul vii.

```
DEFB  +04,multiply          Y1=LN (2**e'),LN X'
                            Y2=LN (2**(e'-1)),LN (2*X')
DEFB  +0F,addition          LD (2**e')*X')      =LN X
                            LN (2**(e'-1)*2*X')  =LN X
DEFB  +38,end-calc          LN X
RET                         Sfîrsit.
```

THE 'REDUCE ARGUMENT' SUBROUTINE (SUBRUTINA 'REDUCERE ARGUMENT')
(Deplasament 39: 'get-argt')

Această subrutină transformă argumentul X al lui SIN X sau COS X într-o valoare V.
Subrutina găseste mai întîi o valoare Y în felul următor:

Y = X/(2*PI) - INT(X/2*PI) + 0,5), unde Y este mai mare sau egal cu -5 dar mai mic decît +5.
Subrutina revine cu:

```
   V = 4*Y       dacă -1<=4*Y<=-1      - cazul i.
sau  V = 2 - 4*y  dacă  1< 4*Y < 2      - cazul ii.
sau  V = -1*Y - 2 dacă -2<=4*Y<=-1      - cazul iii.
```

La fiecare caz, -1<=v<=1 si SIN (PI*V/2) = SIN X.

```
3783 get-argt   RST   0028,FP-CALC          X
                DEFB  +3D,re-stack          X (în forma în virgulă mobilă
                                                desfăsurată)
                DEFB  +34,stk-data          X,1/(2*PI)
                DEFB  +EE,exponent+7E
                DEFB  +22,+F9,+83,+6E
                DEFB  +04,multiply          X/(2*PI)
                DEFB  +31,duplicate         X/(2*PI),X/(2*PI)
                DEFB  +A2,stk-half          X/(2*PI),X/(2*PI),0,5
                DEFB  +0F,addition          X/(2*PI),X/(2*PI)+0,5
                DEFB  +27,int.1C46          X/(2*PI),INT (X/(2*PI)+0,5)
                DEFB  +03,subtract,174C     X/(2*PI)-INT (X/(2*PI)+0,5)=Y
```

Notă: Adunînd 0.5 si luînd INT se rotunjeste rezultatul la cel mai apropiat întreg.

```
                DEFB  +31,duplicate         Y,Y
                DEFB  +0F,addition          2*Y
                DEFB  +31,duplicate         2*Y,2*Y
                DEFB  +0F,addition          4*Y
                DEFB  +31,duplicate         4*Y,4*Y
                DEFB  +2A,abs               4*Y,ABS (4*Y)
                DEFB  +A1,stk-one           4*Y,ABS (4*Y),1
                DEFB  +03,subtract          4*Y,ABS (4*Y)-1-Z
                DEFB  +31,duplicate         4*Y,Z,Z
                DEFB  +37,greater-0         4*Y,Z,(1/0)
                DEFB  +C0,st-meme-0         Mem-0   contine   rezultatul
                                            testului.
                DEFB  +00,jump-true         4*Y,Z
                DEFB  +04,to 37A1,ZPLUS     4*Y,Z
                DEFB  +02,delete            4*Y
```

```
                DEFB  +38,end-calc          4*Y = V  - cazul i.
                RET                         Terminat.
```

Dacă s-a făcut saltul se continuă.

```
37A1 ZPLUS      DEFB  +A1,stk-one           4*Y,Z,1
                DEFB  +03,subtract          4*Y,Z-1
                DEFB  +01,exchange          Z-1,4*Y
                DEFB  +36,less-0            Z-1,(1/0)
                DEFB  +00,jump-true         Z-1
                DEFB  +02,to37A8,YNEG       Z-1
                DEFB  +1B,negate            1-Z
37A8 YNEG       DEFB  +38,end-calc          1-Z = V  - cazul ii.
                                            Z-1 = V  - cazul iii.
                RET                         Terminat.
```

THE 'COSINE' FUNCTION (FUNCTIA 'COS')
(Deplasament 20: 'cos')

Această subrutină tratează functia COS X si redă o 'ultimă valoare' care este o aproximare a lui COS X.
Subrutina foloseste expresia:
COS X = SIN (PI*W/2), unde -1<=W<=1
Derivind W din X subrutina foloseste rezultatul testului obtinut în subrutina anterioară si în acest scop îl stochează în mem-0. Apoi se execută un salt la SINE, subrutină, intrînd la C-ENT, pentru a produce o 'ultimă valoare' a lui COS X.

```
37AA cos        RST   0028,FP-CALC          X
                DEFB  +39,get-argt          V
                DEFB  +2A,abs               ABS V
                DEFB  +A1,stk-one           ABS V,1
                DEFB  +03,subtract          ABS V-1
                DEFB  +E0,get-mem-0         ABS V-1,(1/0)
                DEFB  +00,jump-true         ABS V-1
                DEFB  +06,to 37B7,C-ENT     ABS V-1 = W
```

Dacă nu s-a execu tat saltul atunci se continuă.

```
                DEFB  +1B,negate            1-ABS V
                DEFB  +33,jump
                DEFB  +03,to 37B7,C-ENT     1-ABS V = W
```

THE 'SINE' FUNCTION (FUNCTIA 'SIN')
(Deplasament 1F: 'sin')

Această subrutină tratează functia SIN X si este a treia din cele patru subrutine care folosesc SERIES GENERATOR pentru a produce polinoamele Cebîsev.
Aproximarea pentru SIN X se găseste în modul următor:

i. Argumentul X este redus si în acest caz W=V direct.
De notat că -1<=W<=1, asa cum este necesar pentru ca seriile să fie convergente.
ii. Este format argumentul Z, astfel încît Z=2*W*W-1.
iii. Se foloseste SERIES GENERATOR pentru a reda (SIN (PI*W/2))/W.
iv. In final o simplă înmultire dă SIN X.

```
37B5 sin        RST   0028,FP-CALC          X
```

Se execută pasul i.

```
                DEFB  +39,get-argt          W
```

Se execută pasul ii.De acum înainte subrutina este comună si pentru functia SINE (SINUS) si pentru functia COSINE (COSINUS).

```
37B7 C-ENT      DEFB  +31,duplicate         W,W
                DEFB  +31,duplicate         W,W,W
                DEFB  +04,multiply          W,W*W
                DEFB  +31,duplicate         W,W*W,W*W
                DEFB  +0F,addition          W,2*W*W
                DEFB  +A1,stk-one           W,2*W*W,1
                DEFB  +03,subtract          W,2*W*W-1 = Z
```

Se execută pasul iii, trecînd în SERIES GENERATOR parametrul '6' si cele sase constante cerute.

```
                DEFB  +86,series-06         W,Z
          1.    DEFB  +14,exponent+64
```

```
                DEFB  +E6,(+00,+00,+00)
2.              DEFB  +5C,exponent+6C
                DEFB  ++1F,+0B,(+00,+00)
3.              DEFB  +A3,exponent+73
                DEFB  +8F,+38,+EE,(+00)
4.              DEFB  +E9,exponent+79
                DEFB  +15,+63,+AB,+23
5.              DEFB  +EE,exponent+7E
                DEFB  +92,+0D,+CD,+ED
6.              DEFB  +F1,exponent+81
                DEFB  +23,+5D,+1B,+EA
```

La sfîrsitul ultimei bucle 'ultima valoare' este (SIN (PI*W/2))/W.

Se executà pasul v.

```
                DEFB  +04,multiply          SIN (PI*W/2) = SIN X (sau =
                                            COS X)
                DEFB  +38,end-calc
                RET                         Sfîrsit: 'ultima valoare' =
                                            SIN X sau ('ultima valoare' =
                                            COS X)
```

THE 'TAN' FUNCTION (FUNCTIA 'TANGENTA')
(Deplasament 21: 'tan')

Această subrutină tratează functia TAN X. Această subrutină pur si simplu redă SIN X/COS X, cu depàsire aritmetică dacă COS X = 0.

```
37DA tan        RST   0028,FP-CALC          X
                DEFB  +31,duplicate         X,X
                DEFB  +1F,sin               X,SIN X
                DEFB  +01,exchange          SI X,X
                DEFB  +20,cos               SIN X,COS X
                DEFB  +05,division          SIN X/COS X = TAN X
                                            Se    raportează     depàsire
                                            aritmetică dacă este necesar.
                DEFB  +38,end-calc          TAN X
                RET                         Sfîrsit: 'ultima valoare' =
                                            TAN X.
```

THE 'ARCTAN' FUNCTION (FUNCTIA 'ARCTAN')
(Deplasament 24: 'atn')

Această subrutină tratează functia ATN X si este ultima din cele patru subrutine care folosesc SERIES GENERATOR pentru a produce polinoamele Cebîsev. Ea redă un număr real cuprins între -PI/2 si PI/2, care este egal cu valoarea în radiani a unghiului a cărui tangentă este X.
Aproximarea pentru ATN X este gàsită în felul următor:

i. Valorile lui W si Y sînt gàsite pntru trei cazuri ale lui X, care sînt:

dacă -1<X<1	atunci W=0	& Y=X	- cazul i
dacă 1<=X	atunci W=PI/2	& Y=-1/X	- cazul ii
dacă X<=-1	atunci W=-PI/2	& Y=-1/X	- cazul iii

In fiecare caz, -1<=Y<=1, asa cum este necesarpentru ca seriile să fie convergente.

ii. Argumentul Z este format după cum urmează:

dacă -1<X<1	atunci Z=2*Y*Y-1=2*X*X-1	- cazul i
dacă 1<=X	atunci Z=2*Y*Y-1=2/(X*X)-1	- cazul ii
dacă X<=-1	atunci Z=2*Y*Y-1=2/(X*X)-1	- cazul iii

iii. SERIES GENERATOR este folosit pentru producerea functiei cerute.

iv. In final o simplă înmultire si o adunare va da ATN X.

Se executà faza i.

```
37E2 atn         CALL  3297,RE-STACK        Se foloseste forma în virgulă
                                            mobilă desfàsurată a lui X.
                 LD    A,(HL)               Se aduce exponentul lui X.
                 CP    +81
                 JR    C,37F8,SMALL         Salt înainte în cazul i: Y=X.
                 RST   0028,FP-CALC         X
                 DEFB  +A1,stk-one          X,1
                 DEFB  +1B,negate           X,-1
                 DEFB  +01,exchange         -1,X
                 DEFB  +05,division         -1/X
                 DEFB  +31,duplicate        -1/X,-1/X
```

```
                DEFB  +36,less-0          -1/X,(1/0)
                DEFB  +A3,stk-pi/2        -1/X,(1/0),PI/2
                DEFB  +01,exchange        -1/X,PI/2,(1/0)
                DEFB  +00,jump-true       -1/X,PI/2
                DEFB  +06,to 37FA,CASES   Salt înainte pentru cazul ii:
                                          Y = -1/x   W = PI/2
                DEFB  +1B,negate          -1/X,PI/2
                DEFB  +33,jump            - - - - -
                DEFB  +03,to 37FA,CASES   Salt înainte pentru cazul iii
                                          Y = -1/X    W = -PI/2
37F8 SMALL      RST   0028,FP-CALC        Y
                DEFB  +A0,stk-zero        Y,0
                                          Se continuă pentru cazul i:
                                          W = 0
```

Se execută pasul ii.

```
37FA CASES      DEFB  +01,exchange        W,Y
                DEFB  +31,duplicate       W,Y,Y
                DEFB  +31,duplicate       W,Y,Y,Y
                DEFB  +04,multiply        W,Y,Y*Y
                DEFB  +31,duplicate       W,Y,Y*Y,Y*Y
                DEFB  +0F,addition        W,Y,2*Y*Y
                DEFB  +A1,stk-one         W,Y,2*Y*Y,1
                DEFB  +03,subtract        W,Y,2*Y*Y-1 = Z
```

Se execută pasul iii, trecînd în SERIES GENERATOR parametrul '12' zecimal, si cele 12 constante cerute.

```
                DEFB  +8C,series-0        W,Y,Z
          1.    DEFB  +10,exponent+60
                DEFB  +B2,(+00,+00,+00)
          2.    DEFB  +13,exponent+63
                DEFB  +0E,(+00,+00,+00)
          3.    DEFB  +55,exponent+65
                DEFB  +E4,+8D,(+00,+00)
          4.    DEFB  +58,exponent+68
                DEFB  +39,+BC,(+00,+00)
          5.    DEFB  +5B,exponent+6B
                DEFB  +98,+FD,(+00,+00)
          6.    DEFB  +9E,exponent+6E
                DEFB  +00,+36,+75,(+00)
          7.    DEFB  +A0,exponent+70
                DEFB  +DB,+E8,+B4,(+00)
          8.    DEFB  63,exponent+73
                DEFB  +42,+C4,(+00,+00)
          9.    DEFB  +E6,exponent+76
                DEFB  +B5,+09,+36,+BE
         10.    DEFB  +E9,exponent+79
                DEFB  +36,+73,1B,+5D
         11.    DEFB  +EC,exponent+7C
                DEFB  +D8,+DE,+63,+BE
         12.    DEFB  +F0,exponent+80
                DEFB  +61,+A1,+B3,+0C
```

La sfîrsitul ultimei bucle 'ultima valoare' este:

```
ATN X/X                 - cazul i.
ATN (-1/X)/(-1/X)       - cazul ii.
ATN (-1/X)/(-1/X)       - cazul iii.
```

Se execută pasul iv.

```
                DEFB  +04,multiply        W,ATN X       - cazul i.
                                          W,ATN (-1/X)  - cazul ii.
                                          W,ATN (-1/X)  - cazul iii.
                DEFB  +0F,addition        ATN X - toate cazurile
                DEFB  +38,end-calc
                RET                       Sfîrsit: 'ultima valoare' =
                                          ATN X.
```

THE 'ARCSIN' FUNCTION (FUNCTIA 'ARCSIN')
(Deplasament 22: 'asn')

Această subrutină tratează functia ASN X si redă un număr real cuprins între -PI/2 si PI/2 inclusiv, care este egal cu valoarea în radiani a unghiului al cărui sinus este X.
Astfel dacă Y = ASN X atunci X = SIN Y
Această subrutină foloseste identitatea trigonometrică:

TAN (Y/2) = SIN Y/(1 + COS Y)
pentru a obtine TAN (Y/2) si de aici (utilizînd ATN) Y/2 si în final Y.

```
3833 asn        RST   0028,FP-CALC          X
                DEFB  +31,duplicate         X,X
                DEFB  +31,duplicate         X,X,X
                DEFB  +04,multiply          X,X*X
                DEFB  +A1,stk-one           X,X*X,1
                DEFB  +03,subtract          X,X*X-1
                DEFB  +1B,negate            X,1-X*X
                DEFB  +28,sqr               X,SQR (1-X*X)
                DEFB  +A1,stk-one           X,SQR (1-X*X),1
                DEFB  +0F,addition          X,1+SQR (1-X*X)
                DEFB  +05,division          X/(1+SQR (1-X*X)) = TAN (Y/2)
                DEFB  +24,atn               Y/2
                DEFB  +31,duplicate         Y/2,Y/2
                DEFB  +0F,addition          Y = ASN X
                DEFB  +38,end-calc
                RET                         Sfîrsit: 'ultima valoare' =
                                            ASN X
```

THE 'ARCCOS' FUNCTION (FUNCTIA 'ARCCOS')
(Deplasament 23: 'acs')

Această subrutină tratează functia ACS X si redă un număr real cuprins între zero si PI inclusiv care este egal cu valoarea în radiani a unghiului al cărui cosinus este X.
Subrutina foloseste relatia:
ACS X = PI/2 - ASN X

```
3843 acs        RST   0028,FP-CALC          X
                DEFB  +22,asn               ASN X
                DEFB  +A3,stk-pi/2          ASN X,PI/2
                DEFB  +03,subtract          ASN X-PI/2
                DEFB  +1B,negate            PI/2-ASN X = ACS X
                DEFB  +38,end-calc
                RET                         Sfîrsit: 'ultima valoare' =
                                            ACS X
```

THE 'SQUARE ROOT' FUNCTION (FUNCTIA 'RADACINA PATRATA')
(Deplasament 28: 'sqr')

Această subrutină tratează functia SQR X si redă rădăcina pătrată pozitivă a unui număr real X dacă X este pozitiv, si zero dacă X este negativ. O valoare negativă a lui X dă dreptul la prezentarea A - argument invalid (prin In în subrutina EXPONENTIATION).
Această subrutină tratează operatia rădăcina pătrată ca fiind X**5 si astfel stochează valoarea .5 si trece direct la subrutina EXPONENTIATION.

```
384A sqr        RST   0028,FP-CALC          X
                DEFB  +31,duplicate         X,X
                DEFB  +30,not               X,(1/0)
                DEFB  +00,jump-true         X
                DEFB  +1E,to 386C,LAST      X
```

Se execută saltul dacă X = 0, în caz contrar se continuă cu:

```
                DEFB  +A2,stk-half          X,.5
                DEFB  +38,end-calc
```

si apoi se găseste rezultatul lui X**5.

THE 'EXPONENTIATION' OPERATION (FUNCTIA 'EXPONENTIALIZARE')
(Deplasament 06: 'to-power')

Această subrutină execută operatia binară de ridicare a primului număr, X, la puterea celui de al doilea număr, Y.
Subrutina tratează rezultatul X**Y ca fiind echivalent cu EXP (Y*LN X). Ea redă această valoare numi dacă X este pozitiv si raportează depăsire aritmetică dacă Y este negativ.

```
3851 to-power   RST   0028,FP-CALC          X,Y
                DEFB  +01,exchange          Y,X
                DEFB  +31,duplicate         Y,X,X
                DEFB  +30,not               Y,X,(1/0)
                DEFB  +00,jump-true         Y,X
                DEFB  +07,to 385D,XISO      Y,X
```

Se execută saltul dacă X = 0, în caz contrar se formează EXP (Y*LN X)

```
                DEFB  +25,ln                    Y,LN X
                                                Se dă prezentarea A dacă X
                                                este negativ.
                DEFB  +04,multiply              Y*LN X
                DEFB  +38,end-calc
                JP    36C4,EXP                  Iesire prin EXP pentru a
                                                forma EXP (Y*LN X).
```

Valoarea lui X este zero asa că se iau în considerare cele 3 cazuri implicate.

```
385D XISO       DEFB  +02,delete                Y
                DEFB  +31,duplicate             Y,Y
                DEFB  +30,not                   Y,(1/0)
                DEFB  +00,jump-true             Y
                DEFB  +09,to 386A,ONE           Y
```

Se execută saltul dacă X = 0 si Y = 0, altfel se continuă.

```
                DEFB  +A0,stk-zero              Y,0
                DEFB  +01,exchange              0,Y
                DEFB  +37,greater-0             0,(1/0)
                DEFB  +00,jump-true             0
                DEFB  +06,to 386C,LAST          0
```

Se execută saltul dacă X = 0 si Y este pozitiv, altfel se continuă.

```
                DEFB  +A1,stk-one               0,1
                DEFB  +01,exchange              1,0
                DEFB  ++05,division             Se iese prin 'division' ca
                                                si cum s-ar împărti la zero,
                                                obtinînd 'depăsire
                                                aritmetică'.
```

Rezultatul operatiei trebuie să fie 1.

```
386A ONE        DEFB  +02,delete                -
                DEFB  +A1,stk-one               1
```

Acum se revine cu 'ultima valoare' din stivă ca fiind 0**Y.

```
386C LAST       DEFB  38,end-calc               (1/0)
                RET                             Sfîrsit: 'ultima valoare'
                                                este 0 sau 1.
```

386E - 3CFF Aceste locatii sînt de 'rezervă'. Ele contin +FF. (sînt 1169)

3D00 - 3FFF Aceste locatii contin 'setul caracterelor'. Sînt 8 octeti reprezentati pentru toate caracterele cu codurile cuprinse între +20 (spatiu) si +7F (©).

De exemplu litera 'A' are reprezentarea 00 3C 42 42 7E 42 42 00 si deci are forma:

```
00000000
00111100
01000010
01000010
01111110
01000010
01000010
00000000
```

APENDIX (ANEXA)

BASIC PROGRAMS FOR THE MAIN SERIES (PROGRAMELE BASIC PENTRU SERIILE PRINCIPALE)

Următoarele programe BASIC au fost incluse deoarce ele oferă o bună imagine a modului în care sînt folosite polinoamele Cebîsev pentru a realiza aproximările functiilor SIN, EXP, LN si ATN.

Generatorul seriilor:

Această subrutină este apelată de toate programele 'functie':

```
500 REM SERIES GENERATOR,ENTER
510 REM USING THE COUNTER BREG
520 REM AND ARRAY-A HOLDING THE
530 REM CONSTANTS
540 REM FIRST VALUE IN Z
550 LET M0=2*Z
560 LET M2=0
570 LET T=0
580 FOR I=BREG TO 1 STEP -1
590 LET M1=M2
600 LET U=T*M0-M2+A(BREG+1-I)
610 LET M2=T
620 LET T=U
630 NEXT I
640 LET T=T-M1
650 RETURN
660 REM LAST VALUE IN T
```

```
Z    - valoarea de intrare
T    - valoarea de iesire
M0   - mem-0
M1   - mem-1
M2   - mem-2
I    - contorul pentru BREG
U    - o variabilă temporară pentru T
A(1) la
A(BREG)    - constante
BREG       - numărul de constante ce vor fi folosite
```

Pentru a vedea cum sînt generate polinoamele Cebîsev, se înregistrează pe hîrtie valorile lui U, M1, M2 si T din liniile 550 la 630, trecînd, să zicem, de 6 ori prin buclă, si păstrînd expresiile algebrice de la A(1) la A(6) fără a se substitui valorile numerice. Apoi se înregistrază T-M1. Inmutitorii constantelor A(1) la A(6) vor fi deci polinoamele Cebîsev. Mai exact, înmultitorul lui A(1) va fi $2*T_5(Z)$, al lui A(2) va fi $2*T_4(Z)$ si asa mai departe, pînă la $2*T_1(Z)$ pentru A(5) si în final $T_0(Z)$ pentru A(6).

De notat că $T_0(Z)=1$, $T_1(Z)=Z$ si, pentru $n>=2$, $T_n(Z)=2*Z*T_{n-1}(Z) - T_{n-2}(Z)$.

SIN X

```
 10 REM DEMONSTRATION FOR SIN X
 20 REM USING THE 'SERIES GENERATOR
 30 DIM A(6)
 40 LET A(1)=-.000000003
 50 LET A(2)=0.000000592
 60 LET A(3)=-.000068294
 70 LET A(4)=0.004559008
 80 LET A(5)=-.142630785
 90 LET A(6)=1.276278962
100 PRINT
110 PRINT "ENTER START VALUE IN DEGREES"
120 INPUT C
130 CLS
140 LET C=C-10
150 PRINT "BASIC PROGRAM","ROM PROGRAM"
160 PRINT "--------------","------------"
170 PRINT
180 FOR J=1 TO 4
190 LET C=C+10
200 LET Y=C/360-INT(C/360+5)
210 LET W=4*Y
220 IF W>1 THEN LET W=2-W
230 IF W<-1 THEN LET W=-W-2
240 LET Z=2*W*W-1
250 LET BREG=6
260 REM USE 'SERIES GENERATOR'
270 GO SUB 550
```

```
280 PRINT TAB 6;"SIN";C;"DEGREES"
290 PRINT
300 PRINT T·W,SIN(PI·C/180)
310 PRINT
320 NEXT J
330 GO TO 100
```

NOTE:

i. Cînd se introduce C, acest program calculează si tipăreste SIN C grade, SIN (C+10) grade, SIN (C+20) grade si SIN (C+30) grade. De asemenea tipăreste valorile obtinute folosind programul ROM. Pentru un exemplu de rezultate, încercati să introduceti aceste valori în grade: 0,5; 100; -80; -260; 3600; -7200.

ii. Constantele A(1) la A(6) din liniile 40 la 90 sînt date (diferit de un factor 1/2) în Abramowitz si Stegun - Handbook of Mathematical Functions (Dover 1965) pag.76. Ele pot fi verificate integrînd (SIN (PI*X/2))/X în intervalul U=0 la PI, după ce mai întîi s-a multiplicat cu COS (N*U) pentru fiecare constantă (aducă N=1,2,...,6) si înlocuind COS U = 2*X*X-1. Fiecare rezultat trebuie apoi împărtit cu PI. (Această integrare poate fi realizată prin metode aproximative, ca de exemplu Regula lui Simpson, dacă se lucrează cu un calculator adecvat sau cu un calcultor programabil.

EXP X

```
 10 REM DEMONSTRATION FOR EXP X
 20 REM USING THE 'SERIES GENERATOR'
 30 LET T=0                    (Aceasta face T prima variabilă.)
 40 DIMA(8)
 50 LET A(1)=0.000000001
 60 LET A(2)=0.000000053
 70 LET A(3)=0.000001851
 80 LET A(4)=0.000053453
 90 LET A(5)=0.001235714
100 LET A(6)=0.021446556
110 LET A(7)=0.248762434
120 LET A(8)=1.456999875
130 PRINT
140 PRINT "ENTER START VALUE"
150 IMPUT C
160 CLS
170 LET c=C-10
180 PRINT "BASIC PROGRAM","ROM PROGRAM"
190 PRINT "-------------";"-----------"
200 PRINT
210 FOR J=1 TO 4
220 LET C=C+10
230 LET D=C·1.442695041           (D=C*(1/LN ");EXP C=2**D).
240 LET N=INT D
250 LET Z=D-N                     (2**(N+Z) este acum cerut.)
260 LET Z=2·Z-1
270 LET BREG=8
280 REM USE "SERIES GENERATOR"
290 GO SUB 550
300 LET V=PEEK 32627+256·PEEK 23628+1     (V=(VARS)+1)
310 LET N=N+PEEK V
320 IF N>255 THEN STOP            (STOP cu depăsire aritmetică.)
330 IF N<0 THEN GO TO 360
340 POKE V,N
350 GO TO 370
360 LET T=0
370 PRINT TAB 11;"EXP ";C
380 PRINT
390 PRINT T,EXP C
400 PRINT
410 NEXT J
420 GO TO 130
```

NOTE:

i. Cînd se introduce C acest program calculează si tipăreste EXP C, EXP (C+10), EXP (C+20) si EXP (C+30). De asemenea tipăreste valorile obtinute folosind programul ROM. Pentru un exemplu de rezultate, încercati să introduceti următoarele valori: 0; 15; 65 (cu depăsire la sfîrsit); -100; -40.

ii. Exponentul este testat pentru depăsire si pentru un rezultat zero în liniile 320 si 330. Aceste teste sînt mai simple în BASIC decît în cod masină,

cînd variabila N, spre deosebire de registrul A, nu este limitată la un octet.

iii. Constantele A(1) la A(8) din liniile 50 la 120 se pot obtine integrînd 2**X în intervalul U=0 la PI, după ce mai întîi s-a multiplicat cu COS (N*U) pentru fiecare constantă (adică pentru N=1, 2,...,8) si s-a înlocuit COS U=2*X-1. Fiecare rezultat trebuie apoi împărtit la PI.

LN X

```
 10 REM DEMONSTRARTION FOR LN X
 20 REM USING TEH 'SERIES GENERATOR'
 30 LET D=0                    (Aceasta face D prima variabilă.)
 40 DIM A(12)
 50 LET A(1)=-.0000000003
 60 LET A(2)=0.0000000020
 70 LET A(3)=-.0000000127
 80 LET A(4)=0.0000000023
 90 LET A(5)=-.0000005389
100 LET A(6)=0.0000035828
110 LET A(7)=-.0000243013
120 LET A(8)=0.0001693953
130 LET A(9)=-.0012282837
140 LET A(10)=0.0094766116
150 LET A(11)=-.0818414567
160 LET A(12)=0.9302292213
170 PRINT
180 PRINT "ENTER START VALUE"
190 INPUT C
200 CLS
210 PRINT "BASIC PROGRAM","ROM PROGRAM"
220 PRINT "-------------","-----------"
230 PRINT
240 LET C=SQR C
250 FOR J=1 TO 4
260 LET C=C*c
270 IF C=0 THEN STOP              (STOP cu 'argument invalid')
280 LET D=C
290 LET V=PEEK 23627+256*PEEK 23628+1
300 LET N=PEEK V-128              (N contine e')
310 POKE V,128
320 IF D<=0.8 THEN GO TO 360      (D contine e')
330 LET S=D-1
340 LET Z=2.5*D-3
350 GO TO 390
360 LET N=N-1
370 LET S=2*D-1
380 LET Z=5*D-3
390 LET R=N*0.6931471806          (R contine N*LN 2)
400 LET BREG=12
410 REM USE 'SERIES GENERATOR'
420 GO SUB 550
430 PRINT TAB 8;"LN";C
440 PRINT
450 PRINT S*T+R,LN C
460 PRINT
470 NEXT J
480 GO TO 170
```

NOTE:

i. Cînd se introduce C, acest program calculează si tipăreste LN C, LN (C**2), LN (C**4) si LN (C**8). De asemenea tipăreste valorile obtinute folosind programul ROM.

Pentru un exemplu de rezultate, încercati să introduceti următoarele valori: 1.1; 0.9; 300; 0.004; 1E5 (pentru depăsire) si 1E-5 (STOP ca fiind un 'argument invalid').

ii. Constantele A(1) la A(12) din liniile 50 la 100 pot fi obtinute integrînd 5*LN (4*(X+1)/5)/(4*X-1) în intervalul U=0 la PI, după ce mai întîi s-a înmultit cu COS (N*U) pentru fiecare constantă (adică N=1,2,...,12) si s-a substituit COS U=2*X-1. Apoi fiecare rezultat trebuie împărtit cu PI.

ATN X

```
 10 REM DEMONSTRATION FOR ATN X
 20 REM USING THE 'SERIES GENERATOR'
 30 DIM A(12)
```

```
 40 LET A(1)=-.0000000002
 50 LET A(2)=0.0000000010
 60 LET A(3)=-.0000000066
 70 LET A(4)=0.0000000432
 80 LET A(5)=-.0000002850
 90 LET A(6)=0.0000019105
100 LET A(7)=-.0000131076
110 LET A(8)=0.0000928715
120 LET A(9)=-.0006905975
130 LET A(10)=0.0055679210
140 LET A(11)=-.0529464623
150 LET A(12)=0.8813735870
160 PRINT
170 PRINT "ENTER START VALUE"
180 INPUT C
190 CLS
200 PRINT "BASIC PROGRAM","ROM PROGRAM"
210 PRINT "--------------";"------------"
220 PRINT
230 FOR J=1 TO 4
240 LET B=J*C
250 LET D=B
260 IF ABS B>=1 THEN LET DE=-1/B
270 LET Z=2*D*D-1
280 LET BREG=12
290 REM USE "SERIES GENERATOR"
300 GO SUB 550
310 LET T=D*T
320 IF B>=1 THEN LET T=T+PI/2
330 IF B<=1 THEN LET T=T-PI/2
340 PRINT TAB 8,"ATN";B
350 PRINT
360 PRINT T,ATN B          (sau PRINT T*180/PI,ATN B*180/PI
370 PRINT                  pentru a obtine rezultatul în grade)
380 NEXT J
390 GO TO 160
```

NOTE:

i. Cînd se introduce C, acest program calculează si tipăreste ATN C, ATN (C*2), ATN (C*3) si ATN (C*4).
Pentru un exemplu de rezultate, încercati să introduceti valorile: 0.2; -1; 10 si -100. Rezultatele pot fi găsite mai interesant dacă se converteste pentru a produce gradele, multiplicînd raspunsul din linia 350 cu 180/PI.

ii. Constantele A(1) la A(12) din liniile 40 la 150 se dau (spre deosebire de un factor de 1/2) în Abramowitz si Stegun, Handbook of Mathematical Functions (Dover 1965) pagina 82. Ele pot fi verificate intergrînd ATN X/X în intervalul U = 0 la PI, după ce mai întîi s-a multiplicat cu COS (N*U) pentru fiecare parametru (adică pentru N = 1, 2, ..., 12) si înlocuind COS U = 2*X*X-1. Fiecare rezultat trebuie apoi împărtit cu PI.

O subrutină alternativă pentru SIN X:
Este cinstit să se realizeze întreaga dezvoltare pentru polinoamele Cebîsev si aceasta se poate scrie în BASIC după cum urmează:

```
550 LET T=+(32*Z*Z*Z*Z*Z-40*Z*Z*Z+10*Z)*A(1)
          +(16*Z*Z*Z*Z-16*Z*Z+2)*A(2)
          +(8*Z*Z*Z-6*Z)*A(3)
          +(4*Z*Z-2)*A(4)
          +2*Z*A(5)
          +A(6)
560 RETURN
```

Această subrutină este apelată în locul SERIES GENERATOR si se observă că au aceeasi acuratete.

O subrutină alternativă pentru EXP X:
Intreaga dezvoltare pentru EXP X este:

```
550 LET T=(128*Z*Z*Z*Z*Z*Z*Z*Z-224*Z*Z*Z*Z*Z*Z+112*Z*Z*Z-14*Z)*A(1)
          +(64*Z*Z*Z*Z*Z*Z*Z-96*Z*Z*Z*Z*Z+36*Z*Z-2)*A(2)
          +(32*Z*Z*Z*Z*Z-40*Z*Z*Z+10*Z)*A(3)
          +(16*Z*Z*Z*Z-16*Z*Z+2)*A(4)
          +(8*Z*Z*Z-6*Z)*A(5)
          +(4*Z*Z-2)*A(6)
          +2*Z*A(7)
          +A(8)
```

```
      560 RETURN
```

Dezvoltarea pentru LN X si ATN X, dată în formă algebrică, va fi:

$$
\begin{aligned}
&(2048z^{11}-5632z^{9}+5632z^{7}-2464z^{5}+440z^{3}-22z) * A(1)\\
&+ (1024z^{10}-2560z^{8}+2240z^{6}-800z^{4}+100z^{2}-2) * A(2)\\
&+ (512z^{9}-1152z^{7}+864z^{5}-210z^{3}+18z) * A(3)\\
&+ (256z^{8}-512z^{6}+320z^{4}-64z^{2}+2) * A(4)\\
&+ (128z^{7}-224z^{5}+112z^{3}-14z) * A(5)\\
&+ (64z^{6}-96z^{4}+36z^{2}-2) * A(6)\\
&+ (32z^{5}-40z^{3}+10z) * A(7)\\
&+ (16z^{4}-16z^{2}+2) * A(8)\\
&+ (8z^{3}-6z) * A(9)\\
&+ (4z^{2}-2) * A(10)\\
&+ (2z) * A(11)\\
&+ A(12)
\end{aligned}
$$

THE 'DRAW' ALGORITHM (ALGORITMUL 'DRAW' ('DESENARE'))
Următorul program BASIC ilustrează partea principală a operatiei DRAW (desenare) care a fost folosită la desenarea unei linii drepte. Programul în forma sa prezentă este permis doar pentru linii în care X > Y.

```
      10 REM DRAW 255,175 PROGRAM
      20 REM SET ORIGIN
      30 LET PLOTx=0; LET PLOTy=0
      40 REM SET LIMITS
      50 LET X=255; LET Y=175
      60 REM SET INCREMENT,i
      70 LET i=X/2
      80 REM ENTER LOOP
      90 FOR B=X TO 1 STEP -1
     100 LET A=Y+i
     110 IF X>A THEN GO TO 160
     120 REM UP A PIXEL ON THIS PASS
     130 LET A=A-X
     140 LET PLOTy+1
     150 REM RESET INCREMENT,i
     160 LET i=A
     170 REM ALWAYS ALONG ONE PIXEL
     180 LET PLOTx=PLOTx+1
     190 REM NOW MAKE A PLOT
     200 PLOT PLOTx,PLOTy
     210 NEXT B
```

Un algoritm complet va fi găsit în programul următor, ca o subrutină care va desena o linie ('DRAW A LINE') din ultima pozitie pînă la X,Y.

THE 'CIRCLE' ALGORITHM (ALGORITM 'CIRCLE' (CERC'))
Următorul program BASIC ilustrează cum o comandă CIRCLE realizează cercurile sale.

Mai întîi se calculează numărul cerut de arcuri. Apoi se pregăteste un set de parametrii în 'spatiul de memorie' si în 'stiva calculatorului'.

Apoi se desenează arcurile prin apelări repetate ale subrutinei de desenare linie, care la fiecare apelare desenează o singură linie din 'ultima pozitie' pînă în pozitia 'X,Y'.

Notă: In programul ROM există o linie finală 'de închidere' dar această caracteristică nu a fost inclusă aici.

```
      10 REM A CIRCLE PROGRAM
      20 LET X=127; LET Y=87; LET Z=87
      30 REM How many arcs?
      40 LET Arcs=4*INT(INT(ABS(PI*SQR Z)+0.5)/4+4
      50 REM Set up memory area;M0-M5
      60 LET M0=X+Z
      70 LET M1=0
      80 LET M2=2*Z*SIN(PI/Arcs)
      90 LET M3=1-2*(SIN(PI/Arcs)  2
     100 LET M4=SIN(2*PI/Arcs)
     110 LET M5=2*PI
     120 REN Set up stack; Sa-Sd
     130 LET Sa=X+Z
     140 LET Sb=Y-Z*SIN(PI/Arcs)
     150 LET Sc=Sa
     160 LET Sd-Sb
     170 REM Initialise COORDS
```

```
180 POKE 23677,Sa; POKE 23678,Sb
190 LET M0=Sd
200 REM 'DRAW THE ARCS'
210 LET M0=M0+M2
220 LET Sc=Sc+M1
230 LET X=Sc-PEEK 23677
240 LET Y=M0-PEEK 23678
250 GO SUB 510
260 LET Arcs=Arcs-1; IF Arcs=0 THEN STOP
270 LET MM1=M1
280 LET M1=M1*M3-M2*M4
290 LET M2=MM1*M4+M2*M3
300 GO TO 210

500 REM 'DRAW LINE' from last position to X,Y
510 LET PLOTx=PEEK 23677; LET PLOTy=PEEK 23678
520 LET dx=SGN X; LET dy=SGN Y
530 LET X=ABS X; LET Y=ABS y
540 IF X>=Y THEN GO TO 580
550 LET L=Y; LET B=Y
560 LET ddx=0; LET ddy=0
570 GO TO 610
580 IF X+Y=0 THEN STOP
590 LET L=Y; LET B=X
600 LET ddx=dx; LET ddy=dy
610 LET H=B
620 LET i=INT (B/2)
630 FOR N=B TO 1 STEP -1
640 LET i=i+L
650 IF i<H THEN GO TO 690
660 LET i=i-H
670 LET ix=dx; LET iy=dy
680 GO TO 700
690 LET ix=ddx; LET iy=ddy
700 LET PLOTy=PLOTy+iy
710 IF PLOTy<0 OR PLOTy>175 THEN STOP
720 LET PLOTx=PLOTx+ix
730 IF PLOTx<0 OR PLOTx>225 THEN STOP
740 PLOT PLOTx,PLOTy
750 NEXT N
760 RETURN
```

NOTE ON SMALL INTEGERS AND -65536 (NOTA ASUPRA INTREGILOR MICI SI ASUPRA LUI -65536)

1. Intregii mici n sînt aceia pentru care -65535 este mai mic sau egal cu n care este mai mic sau egal cu 65535. Forma în care sînt tratate este descrisă în 'STACK-BC'. De notat că manualul este inexact cînd spune că al treilea si al patrulea octet contin n plus 131072 dacă n este negativ. Intrucît intervalul lui n este atunci -1 la -65535, cei doi octeti pot să conțină doar n plus 131072 dacă este luat mod 65536; aceasta înseamnă că ei contin 65536. Manualul modifică iesirea. Adevărul este că aceasta nu este o adevărată formă în complement față de doi (asa cum este forma n plus 131072, în anumite circumstante, ar putea fi). Aici, acealsi număr poate fi pus pentru două numere diferite, în concordantă cu octetul semn: de exemplu 00 01 se pune pentru 1 dacă octetul semn este FF; în mod similar FF FF se pune pentru 65535 dacă octetul semn este 00 si pentru -1 octetul semn este FF.

2. Acceptînd că numerele negative sînt date într-o formă specială de 'complement față de doi', trăsătura caracteristică a acestei metode de tratare a numerelor este aceea că ele sînt gata pentru o 'scurtă adunare' fără vreo complementare față de doi mai departe. Ele sînt aduse simplu si stocate direct prin subrutina de adunare. Dar pentru înmultire ele trebuie aduse prin INT-FETCH si stocate după aceea prin INT-STORE. Aceste subrutine complementează numărul în raport cu doi la aducerea sau stocarea lui. Apelul subrutinei INT-STORE este din 'multiply' (după 'short multiplication'), din 'truncate' (după formarea unui 'întreg mic' cuprins între -65535 si 65535 inclusiv), din 'negate'/'abs' pentru 'integer case' si din 'sgn' pentru a stoca 1 sau -1. Apelul subrutinei INT-FETCH se face din PRINT-FP pentru aducerea părtii întregi a numărului cînd este 'small', din 'multiply' de două ori pentru a aduce doi 'small integers', din 'RE-STACK' pentru aducerea unui 'small integer' pentru restocare, din 'negate'/'abs' pentru a aduce un 'small integer' pentru modificare si din FP-TO-BC pentru a aducerea întregului pentru a-l transfera în BC.

Numărul -65536

3. numărul -65536 poate fi reprezentat printr-un 'întreg mic' format ca 00 FF 00 00 00. El este apoi 'număr limitare', unul care cînd este complementat față

de doi depăseste (se compară 80 hex într-un simplu octet sau 7 biti sistem, adică -128 zecimal, care cînd se complementează fată de doi mai dă 80 hex, adică -128 zecimal, cît timp numărul pozitiv 128 zecimal nu poate fi reprezentat în sistem).

4. Cîteva asemănări dintre acestea pot inspira abandonarea în timpul creeri lui 00 FF 00 00 00 în 'truncate'. Este abandonată întrucît aceasta nu ar mai supravietui rutinei INT din care s-a 'trunchiat' o parte. Aceasta doar conduce la greseala INT (-65536) egal cu -1.

5. Dar principala eroare este aceea că acest număr a fost încuviintat să apară din 'adunarea scurtă' a doi întregi mici negativi si apoi a fost pus simplu în stivă ca 00 FF 00 00 00. Sistemul nu poate face fată la acest număr. solutia propusă în 'addition' ('adunare') este acea de a forma imediat toti cei cinci octeti ai formei în virgulă mobilă; această înseamnă prima dată testarea numărului, în jurul octetului 3032, după cum urmează:

```
3032                PUSH  AF              Se salvează octetul semn în A
3033                INC   A               Se transformă orice FF din A
                                          în 00.
3034                OR    E               Se testează toti cei trei
                                          octeti pentru zero.
3035                OR    D
3036                JR    NZ,3040,ADD-STORE   Salt dacă nu este -65536.
3038                POP   AF              Se sterge stiva.
3039                LD    (HL),+80        Se introduce 80 hex în al
                                          doilea octet.
303B                DEC   HL              Se indică primul octet.
303C                LD    (HL),+91        Se introduce 91 hex în primul
                                          octet.
303E                JR    3049,ADD-RSTOR  Salt pentru a seta
                                          indicatorul si iesire.
3040 ADD-STORE      POP   AF              Se readuce octetul semn în A.
3041                LD    (HL),A          Acesta este stocat în stivă.
3042                INC   HL              Este indicată următoarea
                                          locatie.
3043                LD    (HL),E          Se stochează octetul cel mai
                                          putin semnificativ al
                                          rezultatului.
3044                INC   HL              Este indicată următoarea
                                          locatie.
3045                LD    (HL),D          Se stochează octetul cel mai
                                          semnificativ al rezultatului.
3046                DEC   HL              Se mută indicatorul înapoi
3047                DEC   HL              pentru a adresa primul octet
3048                DEC   HL              al rezultatului.
3049 ADD-RSTOR      POP   DE              Se readuce STKEND în DE.
304A                RET                   Sfîrsit.
```

6. Imbunătătirea de deasupra (adică 15 octeti aditionali) cu omiterea octetilor 3223 la 323E inclusiv din 'truncate' trebuie să rezolve problema. Ar fi bine să se poată testa aceasta. Apelul subrutinei INT-STORE nu ar trebui să ducă la stocarea lui 00 FF 00 00 00. In 'multiply' (înmultire) numărul va conduce la depăsire dacă se cere, cît timp 65536 va seta fanionul de transport; asa că se va folosi înmutirea 'lungă'. Cum s-a notat la 30E5, cei 5 octeti începînd de aici pot fi probabil omisi dacă s-a făcut îmbunătătirea de deasupra. 'Negarea' evită stocarea lui 00 FF 00 00 00 tratînd zero separat si returnîndu-l nemodificat. Trunchierea lucrează separat cu -65535, cum s-a notat deasupra. SGN stochează numai 1 si -1.

INDEXAREA RUTINELOR

adresa	rutina	Pag.
RUTINELE DE REPORNIRE si TABELE		
0000	START	6
0008	Eroare	6
0010	Tipărirea unui caracter	6
0018	Colectare caracter	6
0020	Colectarea caracterului următor	6
0028	Calcul	7
0030	Executarea a BC spatii	7
0038	Intrerupere mascabilă	7
0053	ERROR-2	7
0066	Intrerupere nemascabilă	8
0074	CH-ADD+1	8
007D	SKIP-OVER	8
0095	Tabele de simboluri	9
0205	Tabele de taste	10
RUTINELE TASTATURII		
028E	Baleierea tastaturii	11
02BF	Tastatura	12
031C	Repetare tastă	14
031F	K-TEST	15
0333	Decodificarea tastaturii	15
RUTINELE DE DIFUZOR		
03B5	BEEPER	19
03F8	BEEP	21
046E	Tabela de semitonuri	23
RUTINELE DE TRATARE A CASETEI		
04C2	SA-BYTES	24
053F	SA/LD-RET	27
0556	LD-BYTES	28
05E3	LD-EDGE-2	32
0605	SAVE-ETC	33
07CB	Control VERIFY	40
0802	Incărcarea unui bloc de informatii	41
0808	Control LOAD	41
08B6	Control MERGE	44
092C	MERGE Line/Var	47
0970	Control SAVE	48
09A1	Mesajele casetei	49
RUTINELE DE TRATARE A ECRANULUI SI IMPRIMANTEI		
09F4	PRINT-OUT	49
0A11	Tabelul caracterelor de control	50
0A23	Cursor la stînga	50
0A3D	Cursor la dreapta	51
0A4F	Carriage return	51
0A5F	Tipărire virgulă	51
0A69	Tipărire semn de întrebare	51
0AED	Caractere de control cu operanzi	52
0AD9	PO-ABLE	53
0ADC	Rezervare pozitie	54
0B03	Aducere pozitie	54
0B24	Tipărirea oricărui caracter	54
0B7F	Tipărirea tuturor caracterelor	56
0BDB	Setarea octetului atribut	58
0CCA	Tipărire mesaj	59
0C3B	PO-SAVE	60
0C41	Căutare în tabel	60
0C55	Testare scroll (defilare)	60
0CF8	Mesajul 'scroll?'	63
0D4D	Numerele culorii temporare	64
0D6B	Comanda CLS	64
0DAF	Stergerea întregului spatiu al ecranului	65
0DD9	CL-SET	66
0DFE	Defilare	67
0E44	Stergere linii	68
0E8B	CL-ATTR	70
0E9B	CL-ADDR	70
0EAC	Comanda COPY	71
0ECD	COPY-BUF	72
0EDF	CLEAR PRINTER BUFFER	72
0EF4	COPY-LINE	72

0F2C	EDITOR	74
0F81	ADD-CHAR	75
0FA0	Tabelul tastelor de editare	75
0FA9	Tasta EDIT	76
0FF3	Editare cursor jos	77
1007	Editare cursor stînga	77
100C	Editare cursor dreapta	77
1015	Editare DELETE (stergere)	77
101E	ED-IGNORE	77
1024	Editare ENTER	77
1031	ED-EDGE	78
1059	Editare cursor sus	79
1076	ED-SYMBOL	79
107F	ED-ERROR	79
1097	CLEAR-SP	79
10AB	Introducere tastaturà	80
111D	Copierea pàrii inferioare a ecranului	82
1190	SET-HL	83
11A7	REMOVE-FP	83
	RUTINELE EXECUTIVULUI	
11B7	Comanda NEW	84
11CB	Intrarea principalà (Initializare)	84
11DA	RAM-CHECK	84
12A2	Bucla de executie principalà	87
1391	Mesaje de prezentare	90
155D	MAIN-ADD	90
15AF	Canalul initial de informatie	92
15C6	Sirul initial de informatie	92
15D4	WAIT-KEY	92
15E6	INPUT-AD	93
15EF	Tipàrirea principalà	93
1601	CHAN-OPEN	93
1615	CHAN-FLAG	94
162D	Tabelul imagine al codului canalului	94
1634	Fanionul canalului K	95
1642	Fanionul canalului S	95
164D	Fanionul canalului P	95
1652	ONE-SPACE	95
1655	MAKE-ROOM	95
1664	POINTERS	96
168F	Colectarea unui numàr de linie	97
169E	RESERVE	97
16D0	SET-MIN	98
16D4	Readucerea liniei editare	98
16DB	INDEXER	98
16E5	CLOSE (inchiderea) unei comenzi	99
1716	Tabelul 'Trecere prin sir închis'	100
171E	Sir de informatii	100
1736	Comanda OPEN#	101
177A	Tabelul 'Trecere prin sir deschis'	102
1793	Comenzile CAT, ERASE, FORMAT & MOVE	103
1795	Comenzile LIST & LLIST	103
1795	AUTO-LIST	103
17F5	LLIST	104
17F9	LIST	104
1855	Tipàrirea unei întregi linii BASIC	106
18B6	NUMBER	107
18C1	Tipàrirea unui caracter pîlpîitor	107
18E1	Tipàrire cursor	108
190F	LN-FETCH	109
1925	Tipàrirea de caractere într-o linie BASIC	109
196E	LINE-ADDR	111
1980	Compararea numerelor de linie	111
1988	Gàsirea fiecàrei instructiuni	111
19B8	NEXT-ONE	112
19DD	Diferentiere	113
19E5	Refacere	114
19FB	E-LINE-NO	114
1A1B	Tipàrire prezentare si numàr linie	115
	INTERPRETAREA LINIEI BASIC SI A COMENZILOR	116
1A48	Tabelele de sintaxà	116
1B17	Analiza principalà (interpretorul BASIC)	119
1B28	Instructiunea de ciclare	119
1B52	SCAN-LOOP	120
1B6F	SEPARATOR	121
1B76	STMT-RET	121
1B8A	LINE-RUN	121

1B9E	LINE-NEW	122
1BB2	Comanda REM	122
1BB3	LINE-END	122
1BBF	LINE-USE	123
1BD1	NEXT-LINE	123
1BEF	CHECK-END	124
1BF4	STMT-NEXT	124
1C01	Tabelul 'Clasa de comandă'	124
1C0D	Clasele de comandă - 00, 03 & 05	124
1C1E	JUMP-C-R	125
1C1F	Clasele de comandă - 01, 02 & 04	125
1C22	Variabilă în atribuire	125
1C5E	Aducerea unei valori	126
1C79	Asteptare expresie numerică/sir	127
1C96	Setarea culorii permanente (clasa 07)	128
1CBE	Clasa de comandă 0	129
1CDB	Clasa de comandă 0B	129
1CDE	Aducerea unui număr	129
RUTINELE DE COMANDA		130
1CEE	Comanda STOP	130
1CF0	Comanda IF	130
1D03	Comanda FOR	130
1D86	LOOK-PROG	133
1DAB	Comanda NEXT	134
1DDA	NEXT-LOOP	135
1DEC	Comanda READ	135
1E27	Comanda DATA	136
1E39	PASS-BY	137
1E42	Comanda RESTORE	137
1E4F	Comanda RANDOMIZE	137
1E5F	Comanda CONTINUE	138
1E67	Comanda GO TO	138
1E7A	Comanda OUT	138
1E80	Comanda POKE	138
1E85	TWO-PARAM	139
1E94	Găsirea părtii întregi	139
1EA1	Comanda RUN	139
1EAC	Comanda CLEAR	139
1EED	Comanda GO SUB	140
1F05	TEST-ROOM	141
1f1A	Memorie liberă	141
1F23	Comanda RETURN	142
1F3A	Comanda PAUSE	142
1F54	BREAK-KEY	143
1F60	Comanda DEF FN	143
1FC3	UNSTACK-Z	145
1FC9	Comanda LPRINT	145
1FCF	Comanda PRINT	146
1FF5	Tipărire 'carriage return'	146
1FFC	Trimitere informatie	146
2045	Sfîrsitul tipăririi	147
204E	Pozitie tipărire	148
2070	Alterare sir	148
2089	Comanda INPUT	149
21B9	IN-ASSIGN	152
21D6	IN-CHAN-K	153
21E1	RUTINE INFORMATII CULOARE	153
226C	Co-CHANGE	156
2294	Comanda BORDER	157
22AA	Adresa pixelului	157
22C8	Punct	158
22DC	Comanda PLOT	158
2307	STK-TO-BC	159
2314	STK-TO-A	159
2320	Comanda CIRCLE	160
2382	Comanda DRAW	162
247D	Parametrii initiali	168
24B7	Trasare linie	169
EVALUAREA EXPRESIEI		
24FB	SCANNING	172
2530	SYNTAX-Z	173
2535	Baleiere SCREEN$	173
2580	Baleiere ATTR	175
2596	Baleierea tabelului functie	175
26AF	Baleierea rutinelor functie	180
26C9	Baleierea rutinei variabile	181

2734	Baleierea buclei principale	184
2795	Tabelul operatorilor	186
27B0	Tabelul de prioritati	186
27BD	Functia de baleiere (FN)	186
28AB	FN-SKPOVR	191
28B2	LOOK-VARS	192
2951	Argumentul functiei stocate	196
2996	STK-VAR	197
2A52	SLICING	202
2AB1	STK-STORE	204
2ACC	INT-EXP	205
2AEE	DE,(DE+1)	206
2AF4	GET-HL*DE	206
2AFF	Comanda LET	207
2BF1	STK-FETCH	214
2C02	Comanda DIM	214
2C88	ALPHANUM	217
2C8D	ALPHA	218
2C96	Zecimal în virgulă mobilă	218
2D1B	NUMERIC	221
2D22	STK-DIGIT	221
2D28	STACK-A	221
2D2B	STACK-BC	221
2D3B	Intreg în virgulă mobilă	222

RUTINELE ARITMETICE

2D4F	E- format în virgulă mobilă	222
2D7F	INT-FETCH	224
2D8E	INT-STORE	225
2DA2	Virgula mobilă în BC	226
2DC1	LOG (2!A)	226
2DD5	Virgula mobilă în A	227
2DE3	Tipărirea unui număr în virgulă mobilă	228
2F8B	CA=10*A+C	237
2F9B	Pregătire pentru adunare	238
2FBA	Aducerea a două numere	238
2FDD	Deplasare termen adunare	240
3004	ADD-BACK	241
300F	Scădere (03)	241
3014	Adunare (0F)	241
30A9	HL=HL*DE	245
30C0	Pregătire pentru înmultire sau împărtire	246
30CA	Inmultire (04)	246
31AF	Impărtire (05)	252
3214	Trunchiere întreg fată de zero	254
3293	Restocare doi	258
3297	RE-STACK (3D)	258

CALCULUL IN VIRGULA MOBILA

32C5	Tabela de constante	259
32D7	Tabela de adrese	260
335B	CALCULATE	262
33A1	Stergere (02)	264
33A2	Operatie	264
33A9	Testare 5-spatii	265
33B4	Stocare număr	265
33C0	Mutarea unui număr în virgulă mobilă (31)	265
33C6	Stocare literaluri (34)	265
33F7	Omitere constante	267
3406	Locatarea memoriei	268
340F	Aducere din spatiul memoriei (E0 etc.)	268
341B	Stocare constantă (A0, etc.)	268
342D	Stocare în spatiul memoriei	269
343C	EXCHANGE (01)	269
3449	Generator serii (86, etc.)	270
346A	Mărime absolută (2A)	272
346E	Minus unar (1B)	272
3492	Signum (29)	273
34A5	IN (2C)	273
34AC	PEEK (2B)	274
34B3	Număr USR (2D)	274
34DC	Sir USR (19)	274
34E9	TEST-ZERO	276
34F9	Mai mare ca zero (37)	276
3501	NOT (30)	276
3506	Mai mic ca zero (36)	277
350B	Zero sau unu	277
351B	OR (07)	278
3524	Număr AND număr (08)	278

352D	Sir AND număr (10)	278
353B	Comparare (09-0E, 11-16)	279
359C	Concantenarea sirului (17)	281
35BF	STK-PNTRS	281
35C9	CHR$ (2F)	282
35DE	VAL si VAL$ (1D, 18)	282
361F	STR$ (2E)	284
3645	READ-IN (1A)	284
3669	CODE (1C)	285
3674	LEN (1E)	285
367A	Decrementare contor (35)	285
3686	Salt (33)	286
368F	Salt conditionat (00)	286
369B	END-CALC (38)	287
36A0	Modul (32)	287
36AF	INT (27)	288
36C4	Exponential (26)	289
3713	Logaritm natural (25)	291
3783	Reducerea argumentului (39)	294
37AA	Cosinus (20)	295
37B5	SINE (1F)	295
37DA	Tangenta (21)	296
37E2	ARCTAN (24)	296
3833	Arcsinus (22)	299
3843	Arccosinus (23)	299
384A	Radical (extragerea rădăcinii pătrate) (28)	300
3851	Exponentializare (06)	300

ANEXA

Programe BASIC pentru cele mai importante serii:	302
- generator serii	302
- SIN X	302
- EXP X	303
- LN X	305
- ATN X	306
Algoritmul 'DRAW'	308
Algoritmul 'CIRCLE'	308
Notă asupra întregilor mici si -65536	310

Va multumim că ati cumpărat manualul firmei noastre. Acest manual a fost editat si corectat cu toată atentia si presupunem că este corect (dar desigur perfectibil).
ALPHA Ltd. îsi imbunătăteste permanent manualele editate si de aceea vă sintem recunoscători pentru orice sesizare. Vă asteptăm cu orice problemă la sediul firmei si la tel.961/12936